University of Alberta

Distributed Robotic Construction Without Communication

by

Christopher A. C. Parker

A thesis submitted to the Faculty of Graduate Studies and Research in partial fulfillment of the requirements for the degree of **Master of Science**.

Department of Computing Science

Edmonton, Alberta
Fall 2002

National Library of Canada
Acquisitions and Bibliographic Services
395 Wellington Street
Ottawa ON K1A 0N4
Canada

Bibliothèque nationale du Canada
Acquisitions et services bibliographiques
395, rue Wellington
Ottawa ON K1A 0N4
Canada

Your file Votre référence

Our file Notre référence

The author has granted a non-exclusive licence allowing the National Library of Canada to reproduce, loan, distribute or sell copies of this thesis in microform, paper or electronic formats.

L'auteur a accordé une licence non exclusive permettant à la Bibliothèque nationale du Canada de reproduire, prêter, distribuer ou vendre des copies de cette thèse sous la forme de microfiche/film, de reproduction sur papier ou sur format électronique.

The author retains ownership of the copyright in this thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without the author's permission.

L'auteur conserve la propriété du droit d'auteur qui protège cette thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

0-612-81460-2

Canada

University of Alberta

Library Release Form

Name of Author: Christopher A. C. Parker

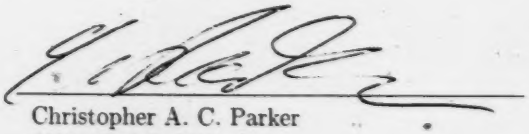
Title of Thesis: Distributed Robotic Construction Without Communication

Degree: Master of Science

Year this Degree Granted: 2002

Permission is hereby granted to the University of Alberta Library to reproduce single copies of this thesis and to lend or sell such copies for private, scholarly or scientific research purposes only.

The author reserves all other publication and other rights in association with the copyright in the thesis, and except as herein before provided, neither the thesis nor any substantial portion thereof may be printed or otherwise reproduced in any material form whatever without the author's prior written permission.



Christopher A. C. Parker
9035 92nd St NW
Edmonton, AB
Canada, T6C 3R3

Date: Oct. 1, 2002

University of Alberta

Faculty of Graduate Studies and Research

The undersigned certify that they have read, and recommend to the Faculty of Graduate Studies and Research for acceptance, a thesis entitled **Distributed Robotic Construction Without Communication** submitted by Christopher A. C. Parker in partial fulfillment of the requirements for the degree of **Master of Science**.

Hong Zhang
Supervisor

Mark Dale

Vadim Bulitko

Ron Kube

Date: Sept. 30, 2002

Go to the ant, thou sluggard;
consider her ways, and be wise:
Proverbs 6:6

Abstract

In this thesis, we present a detailed study of a multiple robot construction system. The inspiration for the algorithm used by our robots was a series of studies done on a particular species of ant, *Leptothorax tuberointerruptus*. The goal of the robots in our system is to cooperate in the construction of a simple nest-like structure. Both a theoretical study of the system and experiments with real robots were conducted. The robots that were used to carry out our experiments were designed especially for this study. Their design and construction is also resented. The predictions of a model that we developed to analyze our system and our experimental results are compared and discussed. This is the first time that actual collective robotic system has been used to build a structure. A modeling strategy that is new to this field is presented in the theoretical treatment of our system.

Acknowledgements

For most of the period in which I was working on this thesis, I was co-supervised by Dr. Max Meng of the Department of Electrical and Computer Engineering at the University of Alberta. Unfortunately, we were unable to schedule a defense of this thesis such that Dr. Meng could attend. I was forced to remove him as my co-supervisor in order for a defense to proceed. Nonetheless, I am indebted to him for the assistance that he gave me with this thesis.

The research described in this thesis was funded in part by grants from NSERC (Government of Canada) and iCORE (Government of Alberta).

To my parents, who encouraged me to expand my
knowledge and who cleaned up most of the resulting messes...

Table of Contents

1 Background **3**
- 1.1 Introduction . . . 3
- 1.2 Traditional Robotics vs. Behaviour Based Robotics . . . 3
- 1.3 Multiple Robot Systems . . . 5
  - 1.3.1 Feasible vs. Optimum Solutions to a Problem . . . 5
  - 1.3.2 Cooperation . . . 6
- 1.4 Stigmergy . . . 7
- 1.5 Difficulties in Collective Systems . . . 8
- 1.6 Collective Construction . . . 9
- 1.7 Summary . . . 9

2 Theory **11**
- 2.1 Introduction . . . 11
  - 2.1.1 A Collective Construction System Based on Ants . . . 11
  - 2.1.2 On the Growth of a Nest . . . 14
  - 2.1.3 Nest Equilibrium . . . 15
- 2.2 Modeling Nest Growth over Time as a Function of the Robot Population . . . 17
  - 2.2.1 Temporal Analysis of a System Containing One Robot . . . 17
  - 2.2.2 Statistical Analysis of an n-Robot System . . . 20
  - 2.2.3 Temporal Analysis of a System Containing n Robots . . . 24
- 2.3 Summary . . . 26

3 Experiments **27**
- 3.1 Introduction . . . 27
- 3.2 Robots . . . 27
  - 3.2.1 Mechanical Hardware . . . 27
  - 3.2.2 Electrical Hardware . . . 31
  - 3.2.3 Robot Software . . . 33
  - 3.2.4 Final Touches . . . 33
  - 3.2.5 Compliance with Model . . . 34
- 3.3 Environment . . . 34
  - 3.3.1 Compliance with Model . . . 35
- 3.4 Running Experiments and Collecting Data . . . 35
  - 3.4.1 Making Measurements . . . 36
- 3.5 Results . . . 36

4 Discussion **40**
- 4.1 Introduction . . . 40
- 4.2 Modeling the Experiment Configurations . . . 40
- 4.3 Comparison of Modeled and Actual Results . . . 42
  - 4.3.1 Discrepancies Between the Model and the Experiments . . . 42
  - 4.3.2 Summary . . . 48
- 4.4 Effect of Robot Population and Plow Threshold on Nest Construction . . . 49

4.4.1 Effect of Population Size 49
4.4.2 Effect of Plow Threshold 49
4.5 Similarities of our System to other Multiple Robot Systems 51
4.6 Next Steps 51
4.7 Summary 52

5 Conclusions **53**
5.1 What Have We Done? 53
5.2 What is Special about this Research? 54
5.3 Why is this Important? 54

Bibliography **55**

A Markov Chain State Transition Probabilities **57**
A.1 Probabilities of Basic Phenomena Occurring in an n-Robot System 57
A.1.1 Probability of Robot-Robot Collision 57
A.1.2 Probability of Pushing Against Nest Wall 58
A.1.3 Basic System Phenomena Summary 58
A.2 Including the Duration of Re-orientation 58
A.2.1 Probability of a Robot in the Finishing or Colliding State Switching to the Plowing State 59
A.3 System State Transition Probabilities 61
A.3.1 Probabilities of Robots in Finishing or Colliding States Entering the Plowing State 61
A.3.2 Probabilities of Robots in the Plowing State Entering the Finishing or Colliding States 62
A.3.3 Probability of the System Remaining in the Same State 62

List of Figures

1.1 The traditional robotics paradigm [15]. This problem solving strategy requires a robot to sense its environment and then create some internal model of the world. Its actions are based around this model. Unfortunately, by the time that a robot can model the world and plan an appropriate action, the world will have changed. Robots that use this approach are not well suited to operation in a dynamic environment. 4

1.2 A behaviour based program for a waiter robot. This robot could carry a tray of drinks about a party. Normally, the robot is controlled by its wander behaviour. When the robot detects an obstacle, the avoid behaviour subsumes the wander behaviour and steers the robot away from a potential collision. Once the robot is pointing away from an obstacle, the wander behaviour regains control of the robot. 4

2.1 A nest of the ant species *Leptothorax tuberointerruptus*. This nest was assembled by ants in a laboratory between two microscope slides separated by squares of card at the corners. The ants built their nest using several construction methods, including simple reactive strategy known as ***blind bulldozing***. This image was obtained from [25]. 12

2.2 A robotic implementation of nest construction by *Leptothorax tuberointerruptus*. Here, robot bulldozers take the place of ants. This is the initial configuration of an experiment. The robots begin in an initial clearing that serves as an initial "nest". The robots will construct their finished nest by plowing the gravel outwards. 13

2.3 A three state finite state machine implementation of the blind bulldozing construction strategy. Normally, a robot is found in the *plowing* state of the machine. A robot in the plowing state will move forward in a straight line. If, while in the plowing state, the force of gravel on a robot's plow exceeds a threshold set in its software, it will enter the *finishing* state. Here, a robot re-orients itself to a random heading and then switches back to the plowing state. If a robot in the plowing state detects a collision with another robot, it will enter the *colliding* state. The behaviour of the colliding state is identical to that of the finishing state. 14

2.4 This diagram depicts a one-robot system. The lone robot never enters the colliding state of its program. Instead, it crosses the nest along the nest's chords in the plowing state. At the endpoint of a chord, the robot pushes back the nest wall somewhat and then enters the finishing state. Here, it re-orients itself and then re-enters the plowing state to head off to push against a new section of the nest wall. 18

2.5 This graph shows the growth of a one-robot system's nest over time as predicted by the equations derived in section 2.2.1. The horizontal line at the top of the graph is the equilibrium size of the nest as predicted in section 2.1.3. Notice how the nest size does indeed approach the predicted equilibrium size. 20

2.6 This figure depicts a three state Markov chain and its transition matrix. Each state has at least one directed edge originating from it. Each edge corresponds to a probability. The sum of all of the probabilities originating from a given state must by one. Thus $P_{3,1} = 1$, $P_{2,2} = 1$ and $P_{1,2} + P_{1,3} = 1$. The elements of the transition matrix are all of the Markov chain's state transition probabilities. The lack of a edge between two states in the graph corresponds to a transition probability of zero. 21
2.7 State diagrams for one, two and three-robot systems. This triangular structure is preserved for all system state diagrams of our system. At most one robot may change state at a time. This makes the maximum number of system state transitions possible from any state five. 22
2.8 Here we see a comparison of the growth predicted by our model between a system containing one robot and a system containing four robots. As one might logically expect, the nest of a four-robot system will grow more quickly than a nest that contains only one robot. Note that both of these nests approach the same equilibrium size. The environmental parameters for both simulations were identical. Only the number of robots participating in construction process was changed. 25

3.1 The toy bulldozer used as a mechanical base for our robots. This toy was readily available and supplied a rugged platform on which to build an intelligent robot. 28
3.2 Robot transmission. The robot's gear box is the dark box in the center and the clutches are the small white objects on either side of it. These clutches made the robots more durable since they protected the gearboxes from unnecessary stress and strain. 29
3.3 Robot chassis and drive system. The robot hull is the large piece of plastic that is seen extending beyond the right of the treads. The hull runs the length of the chassis. The citadel is the plastic box that sits on top of the hull (on the left side of this image). 29
3.4 The finished plow sensor. The thin, vertical board on the right side of the image is the plow itself. A sliding potentiometer is attached to the top of the plow by with a long piece of plastic. The spring which sets the plow sensor's sensitivity can be seen in the upper center of this image. 30
3.5 The on-board computer used in our robots. This computer is based around a Motorola 68HC11, which can be seen as the dark square in the center of the computer board. The column of electrical connectors seen in the upper left of this image are connected to the robot's power manifold. The manifold acts like a power bar for the robot. 31
3.6 This is a behaviour based representation of our robots' control software. The robots have three behaviours: PLOW, RE-ORIENT and SLEEP. The first two behaviours correspond to the three program states. Because the finishing and colliding program states have the same behaviour, we implement them as one behaviour that can be triggered by two different phenomena. The SLEEP behaviour halts a robot when the light level falls below some threshold. 33
3.7 Start state of a four robot experiment. The robots are arranged facing outwards from a round clearing in a field of evenly spread gravel. This image was taken with the overhead camera that was used to record our experiments. 35
3.8 Areas of the nests cleared in experiments with various robot populations. Here we can see the behaviour of the various robot populations with respect to their nests' growths. 37
3.9 Average area cleared for all experimental configurations. This graph depicts the average growth of the various nests for each experimental configuration. . 38

3.10 Radius of nest cleared for all experimental configurations. This graph was produced by assuming that the nests that our robots cleared were circular and solving for the nests radii. This is the type of data which our model produces. 39

4.1 This graph shows the nest growth predicted by our system model for our experimental conditions. The average starting radii for each robot population were used as the initial radii for the modeled curves. 41
4.2 This graph shows both the data from our experiments and our model's predictions of those experiments. 42
4.3 Comparison of area of the nest wall that the model assumes is plowed (left) and the area of the nest wall that is actually plowed (right) 43
4.4 A robot embedded in a soft section of the wall. This robot has plowed deep into the nest wall and will now be prevented from simply rotating to re-orient itself. It will likely take several attempts for this robot to free itself from this soft wall. 45
4.5 Here we see final nests from each of the four different robot populations that we experimented with. Note that by the end of the experiments, the nest walls are relatively uniform in thickness. Also note that the nests are not circular. This means that their walls are longer (and thus weaker per unit length) than our model predicted. 47
4.6 This graph shows the effect that increasing the robots' plow thresholds, F_{p_o}, would have on the nest's equilibrium size. If F_{p_o} is made too large, the robots will end up stuck in the nest wall. 50

A.1 This is a time line representation of a robot leaving the plowing state. In this example, the robot will remain in either the finishing or colliding state for five sampling periods. The robot left the plowing state in the sampling period two and will re-enter it in sampling period seven. 59
A.2 In this example, two robots leave the plowing state. from sampling period four to six there will be two robots in not in the plowing state. 60
A.3 This tree represents all of the different ways that three intervals of length six could overlap each other. Each path from the root of the tree to a leaf represents a mod of overlap. The first index of a leaf of the tree indicates the amount of overlap for that particular mode of overlap. 60

List of Symbols

ρ_o	density of unplowed gravel (pieces per unit area)
F_g	friction due to a single piece of gravel
F_p	force exerted on a robot's plow
F_{p_o}	force threshold of the robot's plow
$r_o > 0$	initial radius of the nest
$r_n \geq r_o$	current radius of the nest
r_r	radius of a robot
w_r	width of a robot's plow
s_o	the speed of a robot
F_{wall}	the force that the wall of the nest presents to a robot
τ	the amount of time that it takes a robot to re-orient itself
t_n	the time until the next incremental progress in an n-robot system

Introduction

Imagine that we have just traveled a few years into the future. A robotic mission to the planet Mars has just departed and will touch down on the Red Planet in about 18 months. There is something different about this mission. Sure, many robotic missions have been sent to Mars, but what makes this mission special is that several dozen robots are being sent at once.

When the robots arrive on Mars, they will operate in a very different manner than other robots that we have sent into space. Most robots that we have sent to other worlds have been specialized individuals that were tailored to their specific missions. The robots that have just been sent will cooperate, much like a swarm of ants. They will build a landing pad, clear a foundation for a space station and engage in other activities as they and mission control see fit. In a sense, we have just sent an all purpose construction crew to pave the way for future missions.

Let us now step back into the present. Researchers *are* designing systems that are based around the cooperation of multiple robots. These systems tend to be more robust than traditional, single robot solutions. Much of the work that is accomplished in our society is due to the cooperation of groups of people. We speak of design teams nowadays instead of individual designers. It should come as no surprise, then, that researchers are looking for ways to make robots cooperate to increase their utility.

In this thesis, we will describe a collective robotic construction system. That is, we will be talking about a group of robots that have been designed to cooperate in the construction of some structure. As we step through this thesis, we will discuss robotics, insects, mechanical design, mathematical modeling and cooperation. Before examining our specific system, a brief survey of other work that has been done in this and related fields of study will be given in Chapter 1. In Chapter 2, a construction algorithm for use with a multiple robot system will be presented and analysed. The analysis will describe the dynamics of the system. We also present a new modeling strategy to predict our system's behaviour that uses Markov chains. The development of the robots that were used to test our distributed construction algorithm will be presented in Chapter 3. This chapter will also describe the experiments that were conducted and briefly present our experimental results. Chapter 4 is devoted

to the discussion of our experimental results and their relation to the model developed in Chapter 2. Discrepancies between our model and our experimental results will be identified and explained. Finally, future directions for research will be outlined. A general conclusion will close the thesis to summarize our findings and their relevance.

Collective robotics is a large and broad field of study. In order to help keep the reader on track, we will motivate our discussion with the hypothetical mission to Mars that has already been described. One of the goals of a group of robots that we might send to Mars might be to build a landing pad for future missions. Anyone who has seen pictures of the surface of Mars would know that the surface of this planet is strewn with rocks. The first stage in the construction of a landing pad would be to clear an area large enough for a pad out of rubble. It is the construction of such a clear area that we will be concerned with in this thesis.

Chapter 1

Background

1.1 Introduction

Before we delve into a robotic algorithm to clear a landing pad on Mars, we will first conduct a brief overview of some of the issues in collective robotics. First, we will explain the difference between the traditional approach to robotics and introduce a robot control strategy known as behaviour based robotics. This discussion will lead into the topic of multiple robot systems. These systems are concerned with feasible solutions to problems. Next, we will discuss the notion of cooperation and present a method for agents to communicate indirectly known as stigmergy. By this point, the reader be wondering "If collective robotic systems are such a good idea, why don't we see them everywhere?" To answer this question, we will present some of the key difficulties encountered when designing these systems. Finally, we will introduce the concept of collective construction.

1.2 Traditional Robotics vs. Behaviour Based Robotics

Robotics has been around for long enough that a significant body of the research done in this field could be classified as *traditional robotics*. Historically, robotic practitioners were concerned mainly with topics such as such as control and world modeling. The high level computational strategy traditionally used in the field of robotics consists of a sequence of operations: sense, model, plan, execute [15, 20, 21]. See Figure (1.1).

The sensing phase takes in information about the world via some sensory apparatus. The modeling phase uses this information to create a model of the world. Next, the planning phase uses this model to determine the appropriate actions necessary to accomplish some aspect of the robot's task. Finally, the execution phase carries out the computed actions. The whole process then repeats. If time and computational resources were not an issue, traditional robotics would represent a valid problem solving strategy. Unfortunately, the real world is complex enough that there is a considerable length of time, even with a powerful computer, between the sensing phase execution phases. By the time that a robot using this

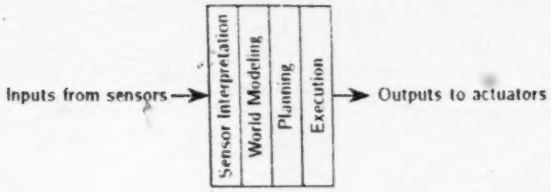


Figure 1.1: The traditional robotics paradigm [15]. This problem solving strategy requires a robot to sense its environment and then create some internal model of the world. Its actions are based around this model. Unfortunately, by the time that a robot can model the world and plan an appropriate action, the world will have changed. Robots that use this approach are not well suited to operation in a dynamic environment.

approach gets around to reacting to the world, the world has changed and the robot's model of it the has become out of date and useless.

In 1986, a new option for robotic control emerged. Touted under the slogan *"the world is its own best model"*, subsumption architecture or behaviour based robotics offered a fast and reliable control scheme that could deal with the complexities of the real world [6]. Behaviour based robotics omits the time consuming modeling and planning phases of traditional robotics. Instead, a behaviour based robot senses the environment and reacts to it immediately by equating certain combinations of sensory inputs with certain actuator outputs.

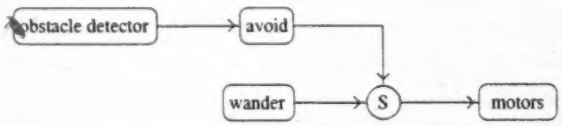


Figure 1.2: A behaviour based program for a waiter robot. This robot could carry a tray of drinks about a party. Normally, the robot is controlled by its wander behaviour. When the robot detects an obstacle, the avoid behaviour subsumes the wander behaviour and steers the robot away from a potential collision. Once the robot is pointing away from an obstacle, the wander behaviour regains control of the robot.

Behaviors simultaneously attempt to utilize a robot's actuators. It is the hierarchy of behaviours in a program determines which behaviour gets control of the robot. We say that a higher level behaviour *subsumes* a lower level behaviour. An example of a simple behaviour based robot program is given in Figure (1.2). In this diagram, the sensors are on the left, the behaviours are in the middle and the actuators are on the right. There are two behaviours: wander and avoid. Wander is normally active and controls the robot's motors. When the obstacle detection sensor detects an obstacle, it triggers the avoid behaviour. The avoid behaviour overrides the wander behaviour and starts to control the motors. When the obstacle detector no longer detects any obstacles, the avoid behaviour turns off and the

wander behaviour regains control of the robot. Note that the wander behaviour never stops issuing commands to the robot's motors. The commands are simply lost while the avoid behaviour is active. We say that the avoid behaviour subsumes the wander behaviour.

One could imagine this robot at a party, carrying a tray with drinks. It would wander about the party, avoiding all of the guests. If we were to implement this robot using a traditional approach, it would likely measure the locations of all of the obstacles within some range. Next, it would determine the best path through all of those obstacles. Finally, it would try to move along that path. Besides all of the unnecessary complexity, guests at the party would likely have moved by the time that the robot would have started to follow its path, making the robot's path infeasible. It would either hit somebody or have to start planning again. The behaviour based approach gives our waiter robot the same functionality with minimal complexity. We could modify our waiter-bot easily, too. By simply adding a weight sensor to its drink tray and a corresponding "return to the kitchen" behaviour to its program whose precedence was above wander but below avoid, the robot would be able to sense when it had run out of drinks and return to the kitchen to restock.

1.3 Multiple Robot Systems

Another characteristic of traditional robotics is that single robots are usually employed to carry out a given task. The last ten years have seen many attempts to build systems of multiple robots that cooperate in to achieve some system goal. Known as *Swarm Intelligence* or *Collective Robotics*, this concept was formally introduced in 1989 [14]. These systems are usually composed of multiple *simple* robots which have little individual intelligence [3, 14, 24]. The idea is to develop a complex group behaviour from simple robots and their interactions [27]. Note the similarity to behaviour based robotics. In a behaviour based robot, it is the interaction of the low level behaviours that produces the robot's overall behaviour. In multiple robot systems, it is the interaction of the individual robots' simple behaviours that produces the complex global system behaviour. The individual robots in a multiple robot system are usually programmed as behaviour based robots. Many applications of collective robotic systems have been studied and implemented. Some have been studied with simulations, many have been implemented with real robots. A few examples are collective transport [17, 23], collective sorting of objects [3, 7, 24], collective foraging [14, 19], collective searching [11] and collective construction [9, 27, 28]. The feasibility of collective robotic systems has been experimentally demonstrated with real robots [7].

1.3.1 Feasible vs. Optimum Solutions to a Problem

Many of the collective robotics experiments to date have been inspired by social insects [17]. Social insects employ a diverse set of collective solutions to many problems [12, 13, 16]. A

key characteristic of these solutions is that they are feasible, not necessarily optimum. A solution to a problem would be optimum with respect to some parameter if it solved that problem better with respect to that parameter than any other solution to that problem. Optimum solutions are nice in theory, but in the real world, they usually require much more time to develop than a feasible solution to the same problem would take to develop. Feasible solutions are attractive because they solve problems in a reasonable amount of time without a lengthy planning period. It doesn't matter if the optimum solution to a problem takes 5 minutes to execute if it requires an additional 3 hours to develop when a feasible solution takes 20 minutes to execute and is relatively obvious. Social insects often employ feasible solutions to problems that would make the most powerful computers grind to a halt [16, 27]. Ants, termites and wasps can construct intricate nests. None of the insects have a blueprint of what they are building, yet they all manage to cooperate to build a useful structure [28]. Most insects usually exhibit a set response to a given stimulus. In addition to this, certain responses seem to override others[1]. If this sounds familiar, it should. It's a subsumption architecture. It can not be proven that insects' brains are structured with subsumption architectures, but they seem to behave that way. The wealth of collective system examples from the insect world, the apparent subsumption architecture of many insects' thought processes and behaviour based robotics' demonstrated feasibility makes the algorithms of social insects tempting examples to follow when developing collective robotic systems.

1.3.2 Cooperation

Up until now, collective robotics and social insects have been discussed. The underlying theme, though, has been cooperation. What does it mean to cooperate? Webster's dictionary defines "to cooperate": *to work or act together toward a common end or purpose.* The *necessity* of working together does not appear in this definition. A distinction between types of cooperation was made in [17]. There are tasks that require cooperation by their very nature (strictly cooperative), and those that can be accomplished by a single agent. Many collective transport tasks are strictly cooperative because the objects being moved are too heavy or awkward for a single robot to manage on its own [23]. Foraging, on the contrary, usually can be carried out by a single robot but is faster when multiple robots cooperate [19].

There are other ways in which one can subdivide the topic of cooperation. How are the cooperating agents organized? In most insect systems, all of the participants are considered equal with no insects in charge [27]. In these systems, each agent works on its part of the system's task. The net effect of all of the agents working on their parts of the task is that the system's task as a whole gets completed. What each agent's part of the task is is often

[1] Ants go about their tasks of foraging and construction but will instantly start to gather the nest's brood and move them to a safe place if the nest is disturbed [12]

dynamic. A wasp may add some paper to one part of its nest, leave to gather more pulp and have another wasp take over construction of that part of the nest. When the first wasp returns, it adds its paper to a different part of the nest. It doesn't matter where the wasp builds as long as it can recognize where building is needed [28]. The wasp need not necessarily be a builder all the time either. It may suddenly start caring for the brood of the nest. Local stimuli often seem to invoke behaviours that subsume others.

In other systems, there are agents whose occupation is the coordination of other agents [16]. Our political systems are a good example of this. We have specialized agents (elected officials) whose only job is to organize the rest of us. As our society grows more complex, the number of organizers has to increase more and more. This means that more resources have to be directed toward the organizers and the process of organization. The net result is that a substantial amount of a hierarchical society's resources get directed toward maintaining the hierarchy itself. Termite nests can be huge. Imagine how large they would have to be if there were termite bureaucrats!

Regardless of the cooperation mechanism, systems of cooperating agents have to have one skill that is unnecessary in a solitary agents: They have to be able to deal with each other. Just because a group of agents are all simultaneously attempting to solve the same problem doesn't mean that they are cooperating. Imagine a crowd of people whose group task is to get everybody through a doorway. All of the people might simultaneously try to get through the door and end up getting stuck. Here, the group would have failed its task. If they had been cooperating, they still would all have the same global goal, but they would work around each other to get the whole group through the door. Social insects (or any social system for that matter) are no different. Unless each insect can work toward the system's goal and accommodate other insects doing the same, no progress will be made. Functioning social systems work because the probability of productive work is greater than the probability of counter productive work. It is not uncommon to see insects occasionally undo to the work of their nest mates [8]. Progress will still be made in an insect colony because on average, the work of the swarm as a whole will be productive. This is a good way of defining cooperation. In a system of multiple cooperating agents, net progress will be made toward the system's goal. If the agents are not cooperating, no progress or negative progress will be made.

1.4 Stigmergy

Most collective robotics work to date has been done using homogeneous agents with no centralized control [7, 10, 17, 28, 29]. How do the agents in these systems know what to do? Each agent reacts to its local environment such that its reactions tend to accumulate toward the goal of the system. The simulated wasps of [28] react to the structure of the nest

being built in such a way that their reactions to it contribute to the nest's construction in an orderly fashion. The nest itself directs the work being done on it. Although the wasps have no blueprints of the nest, they can recognize appropriate places to build on the structure. Because all of the wasps use the same building convention, the nest grows in an orderly fashion. It would be very dangerous to a system such as this if one of the agents began to show some creativity in its building techniques [16], as this might create features on the nest that would confuse other agents. Many insect societies can recognize defective agents and dispose of them before they cause damage [16].

The process of reacting to the environment by modifying it (which then produces new reactions to the modified environment, and so on...) is known as *stigmergy* [7]. Stigmergy literally means "inciting to work" and was identified in 1959 [3]. In systems that employ stigmergy, the environment itself is used as a communication channel. This has many advantages. Messages are left where they are relevant. They also remain after the agent that created the message has left the area. Agents which receive the messages (which may very well be the transmitting agent) do not need to devote resources to the act of listening. They need only pay attention to their environment [7, 3]. Stigmergy has been used in many robotic experiments, including ones that require cooperation in order for the system's task to be completed [14].

1.5 Difficulties in Collective Systems

The advantages of collective robotic systems are many. The robots that they are composed of are usually quite simple This characteristic is especially true when a collective robotic system's robots are compared with a single robot that would be able to complete the collective system's task on its own [3]. Further cost savings due to mass production of robots is also an inherent bonus. Multiple robot systems are also robust through their redundancy [1]. Equipment failures that would render a single robot useless would only remove a single agent from a swarm. With a suitably large swarm, individual agents could be considered expendable.

There are, however, many obstacles to the design of a collective system. Some researchers have regarded their design as a "black art" [10]. The process to design a collective system is currently 1) identify the problem to be solved 2) determine how to solve it in general[2] 3) program robots with the appropriate agent behaviours to create the global behaviour when multiple robots interact. The third step is the problem. As yet it is not known in general how create a local behaviour specification from a global behaviour specification. There have been attempts to create development tools that would create local rules from a global behaviour specification [21, 22], but they are not generally useful. Usually, a researcher has

[2]This becomes the system's global behaviour.

a general idea of how the individual robot controllers should behave and then tunes the global behaviour by tweaking the local rules. There are no hard and fast guidelines as to how to tune the local agent controllers.

Studying these systems is also difficult [20]. Traditional robotic systems can be studied at the microscopic level. Joint angles, velocities and manipulator positions can be predicted easily. Since only one robot is involved, these parameters characterize the performance of the system. In a collective robotic system, the individual velocities and positions of robots may not be known (or relevant). It may not be clear what the relevant system parameters are [18]. One must study the system as a whole [18, 19, 20]. There is a genuine risk of "not seeing the forest for the trees" when analyzing a collective robotic system.

1.6 Collective Construction

Collective construction has received some interest recently [8, 9, 27, 28]. In [27] and [28] a computer simulation of wasps wandering a 3D nest and adding "bricks" to it was described. These are the wasps that were mentioned earlier. Each wasp carried a brick with it and wandered about the surface of a nest. The wasps could see only their immediate surroundings. Their programs consisted of look up tables of local configurations. When a wasp occupied the center of a recognized local configuration, it would place its brick. It would then get another brick and continues to wander the nest as before. By using multiple robot wasps the nest would grow more quickly since the probability of a wasp finding a recognizable local configuration increased with the number of wasps wandering the nest.

The work described in [8] and [9] is of a different vein. Instead of developing an artificial system, the collective construction algorithm used by a species of ant, *Leptothorax tuberointerruptus*, was studied. The ants were supplied with a source of grit with which to construct a nest. The nests of this species are simple, round regions surrounded by a wall of grit with a small opening for an entrance. The colonies could contain up to 500 individuals although the colonies studied in the lab typically contained less than 100 ants. This algorithm is a good candidate for implementation with a collective robotic system. The algorithm of the wasps in [27] and [28] would require mechanically sophisticated robots to implement. Also, since the wasp system has already been implemented in simulation, a robotic implementation would simply be a test of the effects of real world noise on the system's performance, not the algorithm itself.

1.7 Summary

Traditional robotic systems typically employ a single complex robot to solve a problem. Collective robotic systems, on the other hand, use several simple robots to solve the same

problems. The power of these systems comes not from the individual robots but from their interactions. A programming methodology known as behaviour based robotics is often used to program the individual robots because it is computationally efficient and suited to operation in the real world. Many natural systems employ collective solutions to many complex problems. In many of these systems, often insect colonies, the individual members of the systems have no consciousness of the cooperation in which they are participating. One mechanism that allows this sort of cooperation is known as stigmergy. Stigmergy works by allowing the environment direct the agents simply through its configuration. Despite the many advantages to collective robotic systems, they are still difficult to design and analyse. In the next chapter, we will introduce a collective construction algorithm and develop model of its behaviour.

Chapter 2

Theory

2.1 Introduction

In the previous chapter, we discussed some of the topics that form a background to this thesis. Following our example of a mission to Mars, we will now introduce a collective system whose task it will be to clear an area in a field of rubble. Following that, we will speculate on the behaviour of the system over time. The remainder of the chapter will be devoted to the development of a model of our system. With this model, we will return to our hypothesis and discuss it briefly.

2.1.1 A Collective Construction System Based on Ants

The inspiration for our collective construction system is the ant species studied by Franks et al. in [8, 9]. This species, *Leptothorax tuberointerruptus* builds simple, two dimensional nests in rock crevices. Their nests are round regions surrounded by a wall of packed material. In a laboratory setting, the ants were provided with fine grit from which to construct their nests [8].

The ants were observed to use several different strategies to build their nests. The product of the ants' work was a nest that was just the right size for the ant colony that built it. An example of a typical nest is given in Figure (2.1). One of the methods of construction that was observed in the ants was called *blind bulldozing*. When an ant used this strategy, it would push material into the nest wall until the material exerted some force back on the ant. The ant would then turn some amount and head off in a new direction to push more material into another section of the nest wall. The nest wall would be pushed back, too. In this way, ants could enlarge their nest via blind bulldozing. Another construction method involved ants picking up pieces of material and depositing them an appropriate distance from the nest's center. The ants seemed to use the cluster of their nest mates as a template to tell them where to place their building material [8, 9].

The construction strategies of *Leptothorax tuberointerruptus* are suitable for implemen-

Figure 2.1: A nest of the ant species *Leptothorax tuberointerruptus*. This nest was assembled by ants in a laboratory between two microscope slides separated by squares of card at the corners. The ants built their nest using several construction methods, including simple reactive strategy known as ***blind bulldozing***. This image was obtained from [25].

tation by robots in a laboratory setting. These strategies are particularly attractive because they do not require mechanically complex robots. Many insect species such as termites build complex, three dimensional structures. The individual agents in these systems need to be able to negotiate a 3D structure. *Leptothorax tuberointerruptus*, on the other hand, builds a 2D nest on a flat surface. If we restrict the construction strategy of our robotic implementation to blind bulldozing, we need only simple robots. These robots would require two sensors. One sensor would measure the force exerted on a robot by the building material that it was pushing. The other sensor would detect collisions with other robots.

In [8], the ants used fine grit as a building material. Our robot "ants" will be considerably larger than real ants. For our building material, we will use a scaled up version of grit: gravel. The robots will construct a "nest" out of gravel by enlarging a circular clearing in a field of evenly spread gravel. Refer to Figure (2.2) for an image of the initial state of our collective robotic construction system. The robots will construct their final nest using only the blind bulldozing strategy already outlined. Each robot will move in a straight line until either the force of the gravel on its plow reaches a threshold set in the robot's software or the robot detects a collision with another robot in the nest. Once either of these stimuli have been detected by a robot, it will re-orient itself to a random direction. After re-orientation, a robot will resume straight line motion as before.

The behaviour of our robots can be implemented in the form of a finite state machine.

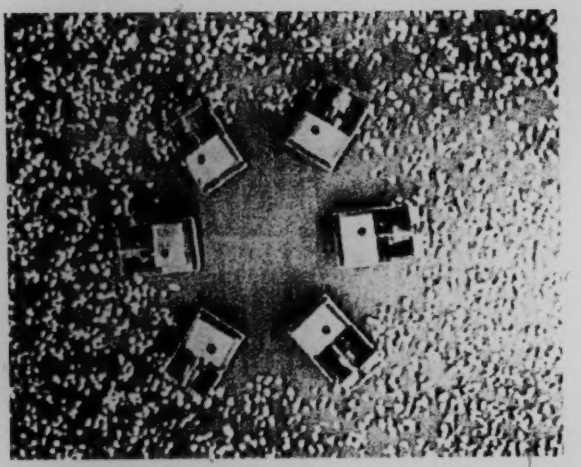


Figure 2.2: A robotic implementation of nest construction by *Leptothorax tuberointerruptus*. Here, robot bulldozers take the place of ants. This is the initial configuration of an experiment. The robots begin in an initial clearing that serves as an initial "nest". The robots will construct their finished nest by plowing the gravel outwards.

A three state finite state machine implementation of robot using the blind bulldozing construction method is given in Figure (2.3). Normally, a robot is found in the *plowing* state of the machine. In the plowing state, a robot moves forward in a straight line. If, while a robot is in the plowing state, the force of gravel on its plow reaches a certain threshold set in its software, it will enter the *finishing* state. Here, a robot re-orients itself to a random heading and then switches back to the plowing state. If a robot in the plowing state detects a collision with another robot, it will enter the *colliding* state. Like the finishing state, a robot in the colliding state will re-orient itself to a random heading and then switch back to the plowing state. Because the behaviours of the finishing and colliding states of a robot's program are identical, one might argue that the finite state machine of Figure (2.3) be reduced to a two state finite state machine. We preserve the finishing and colliding states as distinct states because they signal the occurrence of two different phenomena in our collective robotic system.

A robot enters the finishing state of its program after it has finished a push against the nest wall. Each time that a robot pushes against the nest wall, it will enlarge the nest because the nest wall will have been pushed back some amount, clearing new nest area. The growth of a nest occurs in steps. Each step in a nest's growth starts when a robot pushes against the nest's wall and ends with the robot entering the finishing state of its program. We will call each step of a nest's growth an instance of incremental progress in the nest's construction. The completion of an instance of incremental progress in the nest construction process is signaled whenever a robot enters the finishing program state. The finishing state

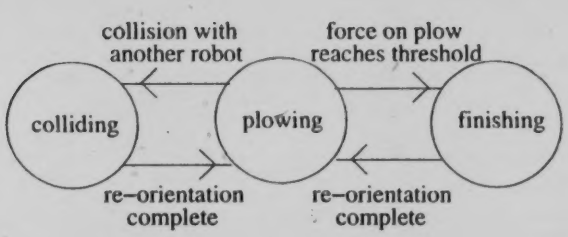


Figure 2.3: A three state finite state machine implementation of the blind bulldozing construction strategy. Normally, a robot is found in the *plowing* state of the machine. A robot in the plowing state will move forward in a straight line. If, while in the plowing state, the force of gravel on a robot's plow exceeds a threshold set in its software, it will enter the *finishing* state. Here, a robot re-orients itself to a random heading and then switches back to the plowing state. If a robot in the plowing state detects a collision with another robot, it will enter the *colliding* state. The behaviour of the colliding state is identical to that of the finishing state.

is so named because it is entered when a robot *finishes* an instance of incremental progress in its nest's construction. We can say that the finishing state of the robots' programs is associated with *productivity* in the nest.

On the other hand, a robot enters the colliding state of its program when it has been interfered with by another robot in the nest. A robot in the colliding state has to turn its attention to obstacle avoidance and away from nest construction. Thus the transition of a robot into the colliding state signals that something *unproductive* has occurred in the construction process.

The actual construction of a nest is accomplished by robots that are in the plowing state of their programs. Robots in both the finishing and colliding states of their programs are temporarily removed from the construction process while they re-orient themselves. A robot in the plowing state is either moving towards the nest wall or is pushing against the it. It is while a robot is moving towards the nest wall that the robot may experience a collision. Once a robot is pushing against the nest wall, it will not detect a collision because its collision sensor is mounted on its plow and will be pointing away from all of the other robots in the nest. Therefore, once a robot begins an instance of incremental progress, it will not be interrupted before the incremental progress is completed.

2.1.2 On the Growth of a Nest

Having outlined the strategy that our robots will use to build their nest, we can turn our attention to the nest itself. Once the robots begin construction, the nest will start to grow. This growth will be achieved by the robots pushing back the nest wall with their plows. How will the nest grow and what will its final size be? In [8, 9], a nest was built that was just the right size for the colony that built it. Since our robots will use a construction strategy

that was seen to be used by ants, it would be logical to hypothesize that the final size of the nest built by our robots will depend on the size of the robot population of the nest.

2.1.3 Nest Equilibrium

Here we will show that the nest will approach some final size. The final size of a nest will be referred to as its *equilibrium* size. To begin our derivation of a nest's equilibrium size, we will make and justify a few assumptions.

1) *Robots always push against the nest wall normal to the nest wall at their point of contact with it.*

This assumption is largely a simplification. We will either need to predict robot trajectories (which would tell us the robots' orientations) or statistically determine some average orientation of a robot when it pushes against the nest wall. Since a robot can, in reality, have any orientation with the nest wall, the mean orientation of a robot while in contact with the nest wall is normal to the the nest wall. Using just one robot orientation simplifies the calculation of the effects of a push against the nest wall. Also, the further that a robot's orientation to the nest wall is from normal to the wall, the thicker the region of packed gravel that the robot would have to push against would be. Thus the effects of a robot pushing against the nest wall not normal it will have less of an effect on the nest's construction than pushes that are normal to the wall.

2) *The initial nest will be circular.*

This is an assumption based on the initial conditions of the nest. The validity of this assumption is dependent on how well we set up our experiments.

3) *The gravel used as building material in our experiments will be spread uniformly about the working surface outside of the nest wall.*

Like (2), this assumption is dependent on our experimental set-up.

4) *The nest will always be circular.*

This assumption follows from the first three. Since the robots orient themselves randomly in the finishing and colliding states, the robots will tend to pay equal attention to each part of the nest wall over time. If the gravel is spread evenly and the robots push against the nest wall normal to it, they will tend to push the nest wall back a constant amount along a radius of the nest regardless of which part of the nest wall that they push against. If the nest starts out as a circle, this shape will be preserved as the nest grows.

5) *The nest wall will have a uniform thickness along its entire length.*

This assumption follows from (4) and (3). If the nest is always circular, then the nest will grow uniformly in every direction. If the gravel is uniformly spread, then the nest wall will sweep up gravel in a uniform fashion as it expands outwards.

Now we turn our attention to the actual calculation of the equilibrium size of the nest. Robots will remain in the plowing state of their programs as long as the force on their plows is below its threshold and the robots do not experience collisions with other robots. We can ignore the colliding state in this development since we are ignoring the time required for a nest to reach its equilibrium size. When robots enter the colliding state, the growth of a nest is simply delayed. Incremental progress on the nest is accomplished every time a robot enters the finishing state of its program. This occurs when the force on a robot's plow reaches some threshold, F_{p_o}. The nest will cease expanding when its wall presents a force equal to F_{p_o} to a robot that pushes against it. That is, a nest's equilibrium size has been reached when Equation (2.1) is satisfied.

$$F_{wall} = F_{p_o} \tag{2.1}$$

Thus we must derive the strength of the nest wall in terms of the nest's size. Since the nest expands uniformly in every direction, the amount of gravel in the nest's wall is a function of the current and initial nest sizes. It is the gravel that was in the area that the robots have cleared that the nest wall is composed of. That is, the amount of gravel in the nest wall is given by Equation (2.2).

$$(\pi r_n^2 - \pi r_o^2)\rho_o \tag{2.2}$$

We assume that each piece of gravel in our experiments is identical. That is, the friction of a pile of gravel is linearly dependent on the number of of pieces of gravel in it and the friction generated per piece of gravel, F_g. Also, our robots' plows have a fixed width, w_r. Thus each robot will only push against a section of the nest wall of fixed width. The force felt by each robot when it pushes against the wall, then, is a function of the wall's strength per unit length and the width of a robot's plow. The force presented to a robot by the wall is given in Equation (2.3).

$$F_{wall} = F_g \rho_o w_r (\pi r_{n_{equilibrium}}^2 - \pi r_o^2)/(2\pi r_{n_{equilibrium}}) \tag{2.3}$$

Now we have an expression for the strength of the nest wall in terms of how a robot experiences it. Combining Equations (2.3) and (2.1), we get Equation (2.4).

$$F_{p_o} = F_g \rho_o w_r (\pi r_n^2 - \pi r_o^2)/(2\pi r_n) \tag{2.4}$$

Equation (2.4) can be rearranged into a quadratic form and solved for the equilibrium nest size in terms of the nest's radius, $r_{n_{equilibrium}}$. This solution is given in Equation (2.5).

$$r_{n_{equilibrium}} = \frac{F_{p_o} + \sqrt{F_{p_o}^2 + F_g^2 \rho_o^2 w_r^2 r_o^2}}{F_g \rho_o w_r} \tag{2.5}$$

This result goes against our initial hypothesis. The nest size *does not* depend on the size of the robot population that created it. In the next sections, we will investigate how a nest grows over time. In a system containing one robot, there is only one agent of progress in the nest. As the number of robots in the system is increased, the amount of work done on the nest per unit time should increase, too.

2.2 Modeling Nest Growth over Time as a Function of the Robot Population

In this section, we will develop a model to predict the growth of the nest of a system that contains n robots. Modeling the behaviour of a multiple robot system has always been a difficult task [21, 22]. Here, we will present a new strategy to model our system.

In many collective systems, there is often a particular instance of the system for which one can develop a model in detail. Often, the other cases of the system present problems that make the direct development of such a model infeasible. For our system, we can model the growth of the nest of a one-robot system over time. Although we may not be able to produce such a model from first principles for the other instances of our system, we can develop a statistical model of our system that includes *all* instances of the system, including a one-robot system. Using such a statistical model, we can make statements about the likelihood of progress occurring in an n-robot system relative to a one-robot system. We will use these statements to indirectly produce a model of the growth of an n-robot nest over time from our model of the growth of a one-robot nest over time.

There is an important feature of our system that allows us to make these sorts of statements. In our system, an instance of incremental progress always occurs in the same fashion, regardless of the population of the system. A nest will always be enlarged when a robot pushes back the nest wall. Two robots will not cooperate in some way to achieve some progress in the nest's construction that a single robot could not have achieved. The progress of our system depends on how many times a robot pushes against the nest wall. If one robot pushes against the wall four times or two robots push against the wall twice each, the growth of the nest will still be the same. It will have grown by four instances of incremental progress. Our system exhibits no strictly cooperative behaviour [17].

2.2.1 Temporal Analysis of a System Containing One Robot

The first stage in our general model is the development of a model of the growth of the nest of a one-robot system. In such a system, the robot will never enter the colliding program state since there are no other robots in the nest to collide with. Instead, it will criss-cross the nest, entering the finishing state after every time that it finishes a push against the nest wall. Refer to Figure (2.4) for a diagram of a one-robot system.

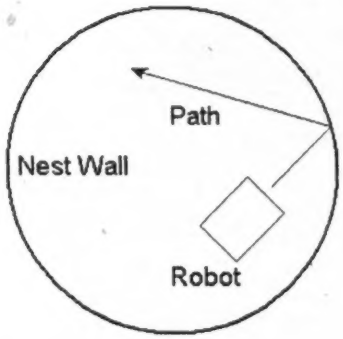


Figure 2.4: This diagram depicts a one-robot system. The lone robot never enters the colliding state of its program. Instead, it crosses the nest along the nest's chords in the plowing state. At the endpoint of a chord, the robot pushes back the nest wall somewhat and then enters the finishing state. Here, it re-orients itself and then re-enters the plowing state to head off to push against a new section of the nest wall.

While calculating the equilibrium size of the nest, it was shown that the nest wall will get stronger (contain more gravel per unit length) as the nest grows in size[1]. We have also discussed the concept of incremental progress in a nest's growth. In order to predict the growth of a nest, we need to know how much it will grow with the completion of the next incremental progress on it and how much time will pass between consecutive instances of incremental progress.

Assume that the lone robot in our system has just returned to the plowing program state from the finishing program state. The nest has a current size characterized by its radius, r_n. The robot *could* be pointing in any direction at this point and it will head off in this direction along some chord of the nest. On average, the robot will head off along the mean chord of the nest that has one of its endpoints at the robot's current position. Thus we need to find the length of the mean chord of a circle with one fixed endpoint. This length can be found easily using numerical integration as $1.27r_n$. The time that it takes the robot to traverse the mean chord of the nest is a function of the robot's speed, s_o. Equation (2.6) gives the time that passes between instances of incremental progress in the nest's construction.

$$t_1 = \frac{1.27r_n}{s_o} \tag{2.6}$$

So, after t_1, the robot will push against the nest wall and expand the nest. How much will the nest expand? This expansion depends on how far the robot will push the nest wall back. How far the robot pushes back the nest wall depends on the wall's strength which is

[1]The fact that the nest has an equilibrium size is proof that the nest walls will strengthen as the nest increases in size since it is the strength of the nest walls that cause the nest's growth to slow down.

a function of how much gravel is in the nest wall. The amount of gravel in the nest wall depends *only* on the nest's current and initial sizes. In order to simplify our calculations, we will assume that all of the gravel that is in the nest wall can be found right at the edge of the cleared nest area. Another way of expressing this assumption is that the wall has zero actual thickness regardless of how much gravel it contains[2]. The amount of gravel in the nest wall was given in Equation (2.2). How far will a robot plow into the nest wall when it pushes against it? Let us assume that this distance is d. The force that is exerted on a robot's plow as it pushes back the nest wall is composed of two terms. The first term is due to the strength of the wall itself, F_{wall}, as calculated in Equation (2.3). The second term is due to the gravel that gets incorporated into the wall as the robot pushes it back. Equation (2.7) shows the force that the robot senses as it pushes the wall back.

$$F_p = F_g \rho_o w_r d + F_{wall} \tag{2.7}$$

The robot will stop pushing the wall back and will enter the finishing state when F_p reaches F_{p_o}. We find the distance that the robot will have pushed the wall back by setting $F_p = F_{p_o}$ and solving for d. At this point, the robot will have just created an approximately rectangular channel into the nest wall of length d and width w_r. The area of this channel is the new area, ΔA, added to the nest due to the instance of incremental progress.

$$\Delta_A = \frac{F_{p_o}}{F_g \rho_o} - \frac{w_r(r_n^2 - r_o^2)}{2r_n} \tag{2.8}$$

The new, *total* nest area is simply the sum of the nest area prior to the incremental progress and the area added by the incremental progress.

$$A_{new} = A_{old} + \Delta_A = \pi r_{n_{old}}^2 + \frac{F_{p_o}}{F_g \rho_o} - \frac{w_r(r_{n_{old}}^2 - r_o^2)}{2r_{n_{old}}} \tag{2.9}$$

The variable of interest has been the nest's radius. We can solve Equation (2.9) and get Equation(2.10) by setting $A_{new} = 2\pi r_{n_{new}}^2$. $r_{n_{old}}$ denotes the radius of the nest prior to the incremental progress and $r_{n_{new}}$, A_{new} denote the radius and area of the nest after the incremental progress has occurred.

$$r_{n_{new}} = \sqrt{r_{n_{old}}^2 - \frac{w_r}{2\pi} r_{n_{old}} + \frac{F_{p_o}}{\pi F_g \rho_o} + \frac{w_r r_o^2}{2\pi} r_{n_{old}}^{-1}} \tag{2.10}$$

Plotting the Growth of a One-Robot Nest

With Equations (2.6) and (2.10), we can plot the growth of the nest of a one-robot system. Initially, $r_n = r_o$ and $t = 0$. Since we are plotting the size of the nest vs. time, we want

[2] Alternatively, we could say that the individual pieces of gravel have mass and friction but zero physical size.

to compute consecutive data points of the form (r_n, t). The next radius of the nest can be found using Equation (2.10) and the current nest radius. The time at which the nest radius will reach its new value can be found using Equation (2.6) and the current nest radius. By repeating this process, we can plot the growth of the nest over time. Figure (2.5) shows a plot of nest growth vs. time for typical values of the various environmental variables.

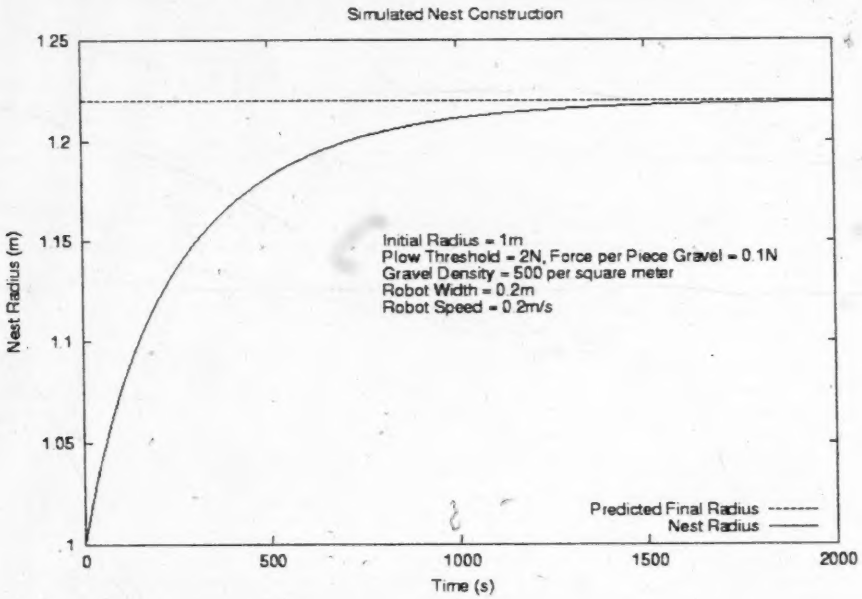


Figure 2.5: This graph shows the growth of a one-robot system's nest over time as predicted by the equations derived in section 2.2.1. The horizontal line at the top of the graph is the equilibrium size of the nest as predicted in section 2.1.3. Notice how the nest size does indeed approach the predicted equilibrium size.

2.2.2 Statistical Analysis of an n-Robot System

We now turn our attention to the development of a statistical description of an n-robot system. We will use this description to generate a temporal description of an n-robot system from the temporal description of a one-robot system that was derived in section 2.2.1.

Markov Chains

Imagine a system that can be represented as a collection of states. The current state of the system changes from time to time. If we cannot specify the next state of the system exactly, but can describe the probability of the system entering specific states based solely on the current state of the system, the system is known as a Markov chain [2]. In a Markov chain, we represent whatever system it is that we are studying as a collection of distinct states

that are linked by state transition probabilities. Thus a Markov chain can be represented by a directed graph with each edge of the graph associated with some state transition probability. For each state of the system, the edges originating from that state must sum to one. A matrix known as the *transition matrix* is associated with a Markov chain. If the states of a Markov chain are numbered 1, 2, ..., n, the dimension of the transition matrix will be $n \times n$. An element of the transition matrix, p_{ab} is the probability of finding the system in state b in the next sampling period given that it is currently in state a. An example of a simple Markov chain and its transition matrix are is given in Figure (2.6).

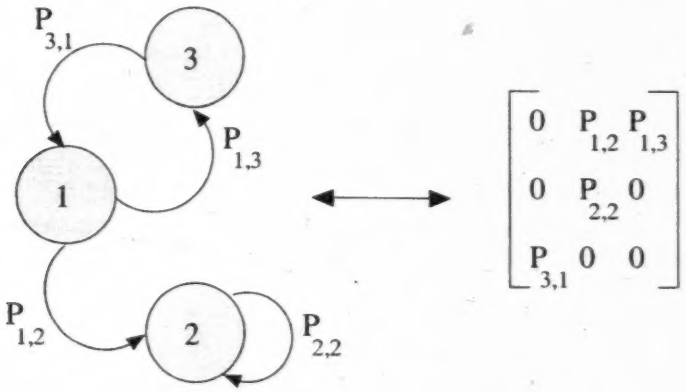


Figure 2.6: This figure depicts a three state Markov chain and its transition matrix. Each state has at least one directed edge originating from it. Each edge corresponds to a probability. The sum of all of the probabilities originating from a given state must by one. Thus $P_{3,1} = 1$, $P_{2,2} = 1$ and $P_{1,2} + P_{1,3} = 1$. The elements of the transition matrix are all of the Markov chain's state transition probabilities. The lack of a edge between two states in the graph corresponds to a transition probability of zero.

We use the transition matrix of a Markov chain to calculate the probability of various phenomena. For example, let us assume for the system in Figure (2.6) that we know the probabilities of finding the system in each of its states in the n^{th} sampling period. What will be the probability of finding the system in each of its states in the $(n+1)^{th}$ sampling period? Refer to Equation (2.11).

$$\begin{bmatrix} P_{1_{n+1}} \\ P_{2_{n+1}} \\ P_{3_{n+1}} \end{bmatrix} = \begin{bmatrix} 0 & P_{1,2} & P_{1,3} \\ 0 & P_{2,2} & 0 \\ P_{3,1} & 0 & 0 \end{bmatrix} \times \begin{bmatrix} P_{1_n} \\ P_{2_n} \\ P_{3_n} \end{bmatrix} \tag{2.11}$$

To find the probability of finding the system in each of its states at time $t_{(n+1)}$, we multiply the probabilities of finding the system in each of its states at time t_n by the system transition matrix. We can continue to repeat this process over and over to find the probability of finding the system in each of its states at time $t_{(n+2)}$, $t_{(n+3)}$, ..., $t_{(n+m)}$. For a certain class of Markov chain, the probability of finding the system in each of its states converges to a constant as m approaches infinity regardless of the initial probabilities

of finding the system in each of its states. This constant probability vector is known as the system's *stationary state probability function*, and can be interpreted as the general probability of finding the system in each of its states. The stationary state probability function can be found using any number of methods, none of which will be discussed here. Refer to [26] for a more thorough treatment of Markov chains.

We will decompose our n-robot system into a finite set of states. Then, we will identify the system state change probabilities. Next, we will calculate our system's stationary state density function. Finally, we will use the system's stationary state density function and the system's transition matrix to calculate the probability of incremental progress occurring in an n-robot system.

A Markov Chain Representation of an n-Robot System

The first step in developing a Markov chain for our collective construction system will be to identify its states. We will borrow an idea from [20]. We will describe the states of our system with an ordered triple, (x, y, z). For a given system state, x is the number of robots in the plowing program state, y is the number of robots in the finishing program state and z is the number of robots in the colliding program state. It should be obvious that for an n-robot system, $x + y + z = n$ and that x, y and z are all integers greater than or equal to zero. The number of states in an n robot system is given by Equation (2.12). System state diagrams of one, two and three-robot systems are given as examples in Figure (2.7).

$$\text{number of states in a system containing } n \text{ robots} = (n+1)(n+2)/2 \tag{2.12}$$

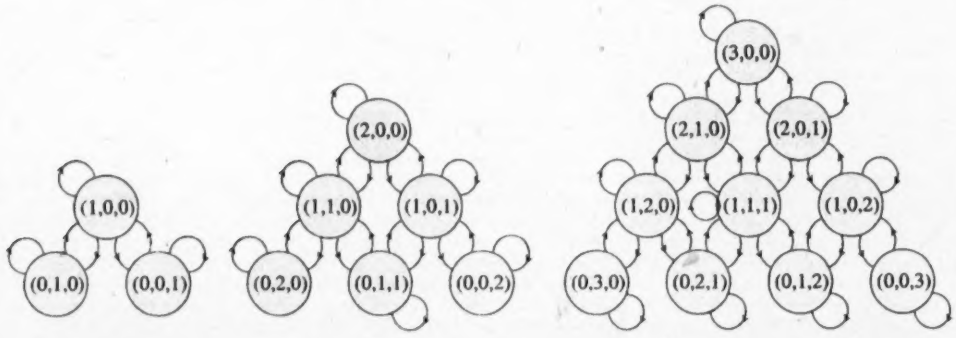


Figure 2.7: State diagrams for one, two and three-robot systems. This triangular structure is preserved for all system state diagrams of our system. At most one robot may change state at a time. This makes the maximum number of system state transitions possible from any state five.

System State Transition Probabilities

Here, we calculate the elements of our system's transition matrix. In practice, it is possible for any number of robots in our system to change program state simultaneously. If we were to allow our model the same freedom, we would have to calculate a state transition probability from every state of our system to every state of our system. If our system were to have m states, we would have to calculate m^2 probabilities. For an n-robot system, this would correspond to $0.25n^4 + 1.5n^3 + 3.25^2 + 3n + 1$ probabilities! Clearly the number of possible system state transitions must be limited in some way to prevent our model from becoming too computationally intensive. We restrict the freedom of the model by allowing only one robot to change program state at a time. From a generic state of our system model there will be at most five possible state transitions. The number of robots in the finishing program state could increase or decrease, the number of robots in the colliding program state could increase or decrease or none of the robots could change program state. Of course, there are system states from which there are less than five state transitions. For example, if the number of robots in the plowing program state in a certain system state is zero, there will be no system state transition to a system state with more robots in the finishing or colliding program states.

All of the system state change probabilities can be calculated for any instance of our collective construction system. A full derivation of these probabilities can be found in Appendix A.

System Stationary State Density Function

It can be shown that the stationary state density function always exists for our system[3]. The stationary state density function of our system tells us the probability of finding the system each of its states. Earlier in this chapter, we equated the phenomenon of a robot entering the finishing program state with incremental progress occurring in its nest's construction. From Figure (2.3), we can see that the finishing program state is only accessible from the plowing state. The probability of a robot entering the finishing program state is is proportional to the number of robots likely to be found in the plowing program state. It is this information that the system stationary state density function tells us. In the next section, we will use the stationary state density function and the transition matrix of our system's Markov chain to calculate the probability of incremental progress occurring in any instance of our system.

Probability of Incremental Progress in an n-Robot System

This is the final section of the statistical analysis of our system. We can calculate the probability of finding our system in any of its states. These probabilities are found as the

[3]The states of our system are all ergodic [26]

stationary state density function of the Markov chain representation of our system. From each system state, we can calculate the probability of incremental progress occurring in the next sampling period. That is, we can calculate the probability that the system will transition from a given state into a system state such that the number of robots in the finishing program state has increased. This set of probabilities (one for each state of the system) is a set of the elements of the Markov chain's transition matrix. Referring to Figure (2.7), these probabilities correspond to the system changing to a state that is immediately to the lower left of the current state.

We define P_n be the probability that incremental progress will occur in the next sampling period in an n-robot system. We can calculate P_n with Equation (2.13). The first term in the sum is the probability of finding the system in a particular state. The second term in the sum is the probability of a robot entering the finishing program state in the next sampling period. If $(i-1, j+1, k)$ is not a state of the system, $P((i-1, j+1, k)|(i, j, k))$ is defined as zero.

$$P_n = \sum_{(i,j,k) \in n\text{-robot system}} P(i,j,k) \times P((i-1,j+1,k)|(i,j,k)) \tag{2.13}$$

To better illustrate how this equation is used, an example of its application to a two-robot system is given in Equation (2.14).

$$\begin{aligned} P_2 = \; & P(2,0,0) \times P((1,1,0)|(2,0,0)) \\ + \; & P(1,1,0) \times P((0,2,0)|(1,1,0)) \\ + \; & P(1,0,1) \times P((0,1,1)|(1,0,1)) \end{aligned} \tag{2.14}$$

Equation (2.13) is defined for all possible instances of our system. Thus for any instance of our collective construction system, we can calculate the probability of incremental progress occurring in the next sampling period. In the next section, we will use this equation to predict the growth of the nest of an n-robot system over time.

2.2.3 Temporal Analysis of a System Containing n Robots

We are now in a position to calculate the growth of an n-robot nest over time. When we calculated the growth of a one-robot nest over time, we calculated the time between the instances of incremental progress in the nest and the effect of each instance of incremental progress on the nest. The method to calculate the effect of incremental progress in the nest has not changed. The effect of the m^{th} push against the nest wall on the nest will be the same regardless of whether one robot pushed against the nest wall m times or m robots each took turns pushing against the nest wall. Given the current and initial sizes of the nest, the nest size after the next instance of incremental progress is given by Equation (2.10). All that changes with multiple robots is the time that passes between instances of incremental progress.

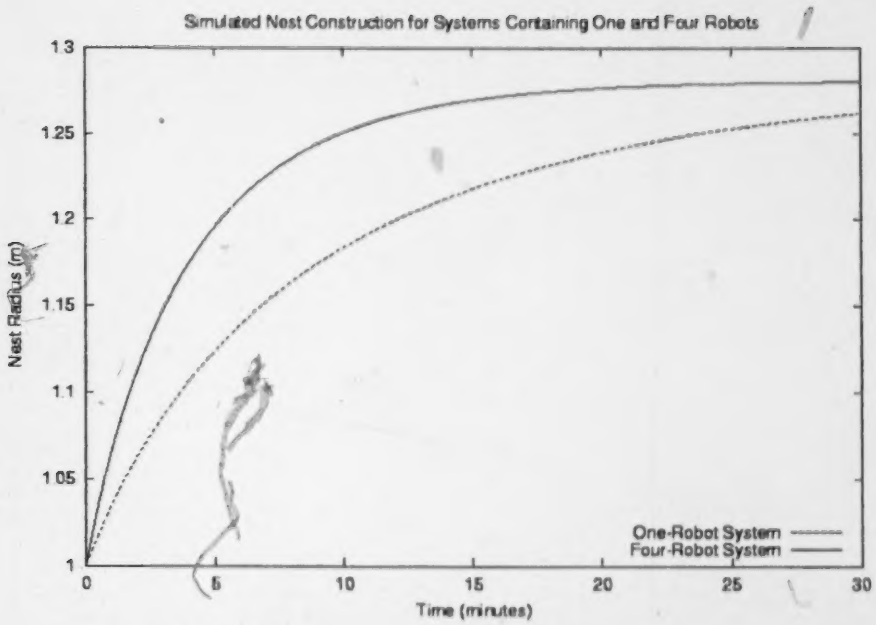


Figure 2.8: Here we see a comparison of the growth predicted by our model between a system containing one robot and a system containing four robots. As one might logically expect, the nest of a four-robot system will grow more quickly than a nest that contains only one robot. Note that both of these nests approach the same equilibrium size. The environmental parameters for both simulations were identical. Only the number of robots participating in construction process was changed.

We begin with a final assumption. Suppose that we have two events, A and B with probabilities of occurring $P(A)$ and $P(B)$. We make the approximation that B will happen in $\frac{P(A)}{P(B)}$ of the time that A will. In other words, if B is twice as likely to occur as A, then we assume that B will occur, on average, in half of the time that A will. In section 2.2.2, we laid out a method to calculate the probability of incremental progress occurring in an n-robot system. Let our two events be incremental progress occurring in a one-robot system and an n-robot system: P_1 and P_n. We can calculate the time until the next incremental progress occurs in a one-robot system, t_1 with Equation (2.6). Using our approximation, we can calculate the time until the next incremental progress occurs in an n-robot system, t_n, using Equation (2.15).

$$t_n = \frac{P_1}{P_n} \times t_1 \tag{2.15}$$

Now we have all of the tools necessary to plot the growth of an n-robot system over time. First, calculate the time until the next instance of incremental progress in a one-robot system. Next, calculate the probabilities of incremental progress occurring in the next sampling period in a one-robot system and in an n-robot system using a Markov chain representation of the systems. Calculate the time until the next instance of incremental progress in an n-robot system using Equation (2.15). Finally, calculate the new nest size using Equation (2.10). Repeat this process to obtain consecutive data points. Figure (2.8) shows the simulated growth of nests containing one and four robots.

2.3 Summary

In this chapter, we introduced a collective construction algorithm to clear an area out of a field of rubble. Our algorithm was inspired by studies done on a particular species of ant. We formulated that algorithm as a controller for a robot bulldozer. Next, we hypothesized that the size of the nest that would be built by our robots would depend on the number of robots taking part in the construction process. After deriving the equilibrium state of our system, we found that this hypothesis was, in fact, wrong. We next hypothesized that a larger group of robots would build a nest faster than a smaller group of robots. Thus we became interested in the growth of an n-robot nest over time. We introduced a new technique to model the behaviour of a non-strictly cooperative system of robots that involved Markov chains and applied it to our system. The addition of robots to a system will increase the rate of nest construction rate but not the final size of the nest produced. In the next chapter, we will describe the experiments that we conducted to experimentally determine the growth of an n-robot system.

Chapter 3

Experiments

3.1 Introduction

We will now introduce the robots that were used to implement our collective construction system. These robots were custom designed for this thesis. A mechanical and electrical description will be provided. We will also discuss the robots' software. Following this, we will describe our experimental environment and the experiments that we conducted. Before closing the chapter, we will briefly present the results of our experiments.

3.2 Robots

The robots that were used in our experiments were designed and assembled especially for this project. What follows is a description of their design.

3.2.1 Mechanical Hardware

When developing robots, there are usually two options for a researcher. The first is to customize every aspect of the robots' design. The advantage of this approach is that researcher has a very high level of control over how well the robots conform to any assumptions made by system models[1]. The disadvantage of this approach is that the design phase of the research may become prohibitively long and expensive. The other option is to modify an existing platform to suit the project's needs. The advantage of this approach is that the design phase will usually be relatively fast and inexpensive since much of the low level design work will have already been taken care of by the designers of the adopted platform. For this thesis, the latter approach was taken. A toy bulldozer was used as the mechanical basis of the robots. See Figure (3.1) for a picture of this toy.

This toy was chosen for several reasons. The first of which was cost. One of the advantages claimed by researchers of collective robotic systems is that the individual robots of such systems are relatively inexpensive and therefore expendable when compared with the

[1] Of course, it may be impossible to conform to assumptions that over simplify parts of the system

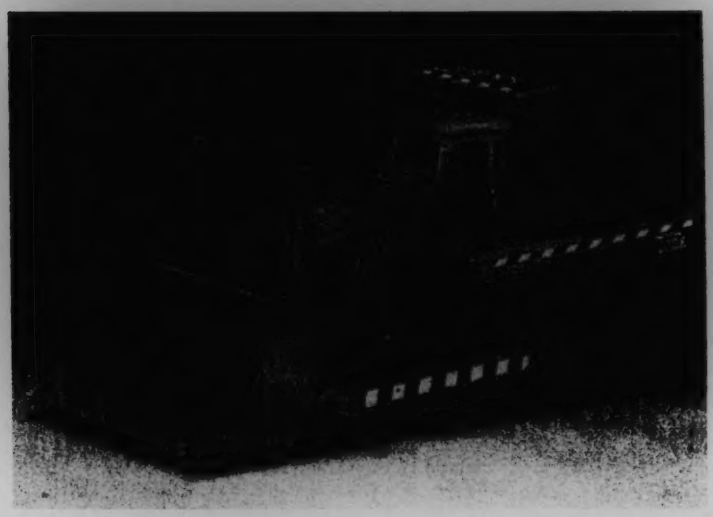


Figure 3.1: The toy bulldozer used as a mechanical base for our robots. This toy was readily available and supplied a rugged platform on which to build an intelligent robot.

equivalent single robot solution for a given task [3]. It is important to practice what one preaches. The toy bulldozer cost approximately $35.00. It featured a relatively light weight chassis employing a tracked, differential drive system. The motors were designed to run off of 6 volts (4 D-cell batteries). The plow on the front of the toy was also motorized, allowing it to be raised or lowered. Clutches were (Figure (3.2)) included on the all of the toy's drive shafts to protect the gearboxes from being damaged in the event that the tracks or the plow became stuck/jammed.

Originally, the plow raising mechanism was to be used in our robots. In the end, this was abandoned as it added no significant functionality to our robots. The first stage in converting the toy into a robot was to remove all but the two bottom pieces of the chassis and its drive system. The lower of the two pieces will be referred to as the robot hull and the piece at the rear that sits on top of the hull as the citadel. Collectively, the two pieces will be referred to as the robot chassis. See Figure (3.3).

All of the electronics that controlled the toy's functions were housed in the upper portions of the toy that were removed. The speaker that produced sound effects in the toy was attached to the hull and was also removed. In order to attach various electronic circuits and mechanical assemblies to the chassis, thin wooden decks were fastened to the exposed upper surfaces of the hull and citadel.

Plow Sensor

The primary sense of the robots is their ability to measure the force that the gravel that is being plowed exerts on the robot. This capability was implemented on the robots with

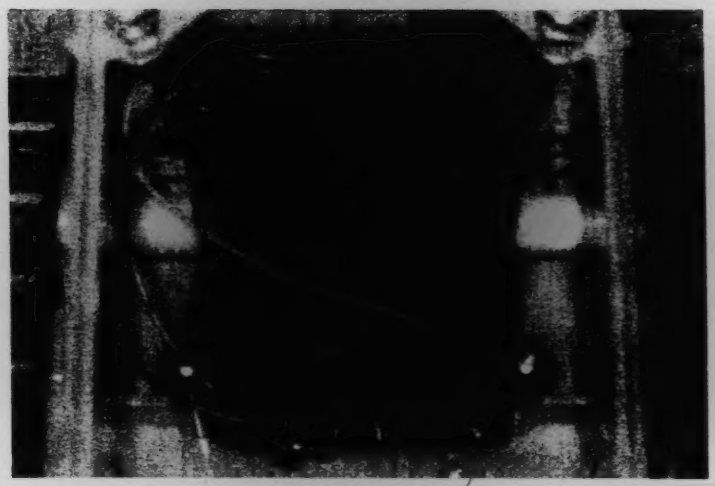


Figure 3.2: Robot transmission. The robot's gear box is the dark box in the center and the clutches are the small white objects on either side of it. These clutches made the robots more durable since they protected the gearboxes from unnecessary stress and strain.

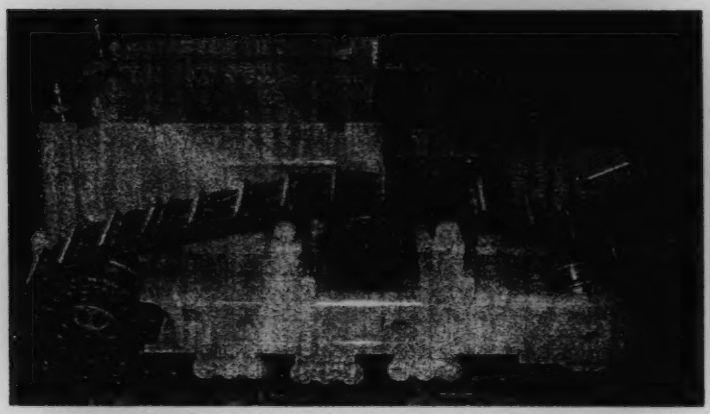


Figure 3.3: Robot chassis and drive system. The robot hull is the large piece of plastic that is seen extending beyond the right of the treads. The hull runs the length of the chassis. The citadel is the plastic box that sits on top of the hull (on the left side of this image).

a force sensitive plow. There were several ways in which such a plow could have been constructed. Because the robot would be relatively slow moving, the force on the plow would change slowly with time. The sensor, then, must be useful at very low frequencies. One simple way to build such a sensor would be to allow a force on the plow to cause some displacement of a spring loaded mechanism. A change in the displacement of the plow would correspond to a force on the plow via the spring. Changing the bias point of the spring would change the sensitivity of the plow sensor. All that would remain would be to measure the mechanism's displacement. This could be done by attaching the mechanism to the wiper of a sliding potentiometer. With the fixed terminals of the potentiometer connected to some reference voltage and ground, the voltage at the wiper of the potentiometer would increase from ground to the reference as the force applied to the plow increased.

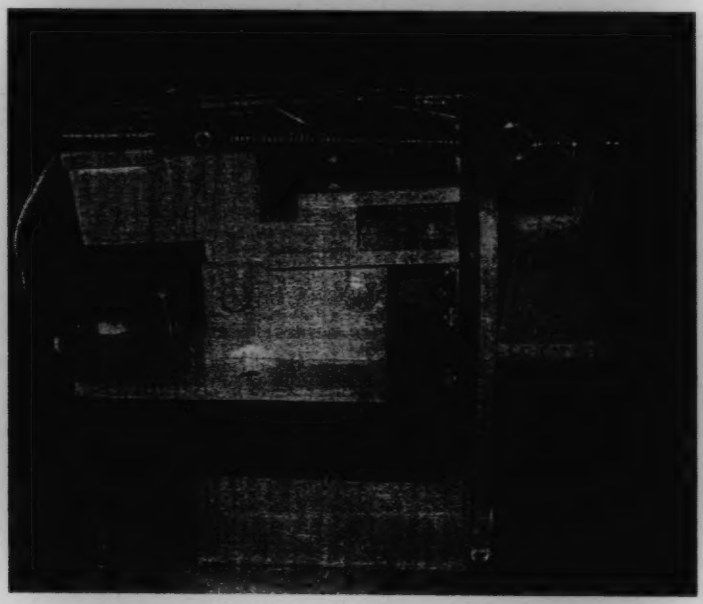


Figure 3.4: The finished plow sensor. The thin, vertical board on the right side of the image is the plow itself. A sliding potentiometer is attached to the top of the plow by with a long piece of plastic. The spring which sets the plow sensor's sensitivity can be seen in the upper center of this image.

In the final design, the plow was hinged with the hinge's pin parallel to the robot's axles. Instead of having the whole plow move backwards, its bottom edge swings back as force is applied. The portion of the plow below the hinge was made as long as possible to provide the maximum amount of plow displacement horizontally with the minimum amount of vertical displacement. The spring was placed with one end attached to to the plow above the hinge and the other end attached further back on the robot. Thus the spring pulls the top of the plow back and the bottom of the plow - the part of the mechanism that actually

plows - forward. In this manner the spring causes the plow to oppose the force applied to the front of the robot. A piece of plastic connects the top edge of the plow to a sliding potentiometer's wiper.

The spring was fastened to the plow using an eye bolt. The eye bolt was used to adjust the tension in the spring and thus the plow's sensitivity. At the bottom edge of the plow a brush-like sweep was attached. This sweep ensured that as the bottom edge of the plow swung back and up, the plow would maintain contact with the surface on which the robot was traveling. Finally, the plow mechanism (plow, spring assembly, potentiometer, etc) was mounted in such a fashion that the entire mechanism could be adjusted vertically. The plow mounting was fastened to the deck that sat on the front half of the robot's hull. The entire assembly was painted the same colour as the robot chassis. The finished plow can be seen in Figure (3.4).

3.2.2 Electrical Hardware

The robots used for this thesis needed only the most rudimentary computational hardware. The computer chosen for the robots is a single board computer manufactured by New Micros Inc. It is identified as the NMIY-0020. See Figure (3.5).

Figure 3.5: The on-board computer used in our robots. This computer is based around a Motorola 68HC11, which can be seen as the dark square in the center of the computer board. The column of electrical connectors seen in the upper left of this image are connected to the robot's power manifold. The manifold acts like a power bar for the robot.

This computer employs a Motorola 68HC11 microcontroller as its CPU. and operates using the FORTH programming language as its operating system. Although the 68HC11 displays an impressive array of features, only two of them were used on our robots. The first is an ADC (analog to digital converter). This peripheral converts a voltage presented to the processor into a number based on its magnitude. The ADC allows the computer to measure

continuous phenomena. The other peripheral used was the timer module. The timer was used to generate two PWM (pulse width modulation) signals to control the speed of our robots' motors.

A readily available H-bridge circuit was used to interface a robot's motors with its on-board computer. The computer was mounted on top of the deck that sits on top of the citadel. The motor interface was mounted upside down on the bottom of this deck. In this way, the interface sat conveniently over the motors where the two components were easily connected. The connections between the computer board and the interface were routed through a hole drilled in the deck.

A separate power supply module was not necessary for our robots as the computer board also contained power regulation circuitry. The computer circuit board also featured a small prototyping area for the user to add any custom circuitry. This area was used to build a power manifold - essentially a power bar for each robot. See Figure (3.5) This manifold could supply electrical power directly from a robot's battery or regulated at 5V from the power circuitry of the computer board. The various sensors and the motor interface were supplied with power via the manifold.

The plow sensor was connected to one of the computer's ADC channels. The motor interface's control lines were connected to the PWM channels (to set motor speed) and to regular output lines (to set motor direction). There were two more sensors on each robot that have yet to be discussed. The first was the collision sensor. Our construction algorithm requires the robots to be able to detect collisions with other robots and react appropriately. This sense was implemented on our robots with a bumper sensor mounted on the front of the plow. Two contact switches supported a plastic bar held in front of the robot. The bar was mounted high enough that no gravel would touch it. The use of two switches mounted at opposite ends of the bar allowed a robot to sense which side of it a collision occurred on. Each robot had wooden panels mounted all around it at the height of the bumper so that from any direction of attack, a robot would present a surface to any colliding robot such that the colliding robot's bump sensor would be triggered. The final sensor was a light sensor mounted on the top of each robot. The use of this sensor will be explained later. It was mounted directly on the computer board and interfaced with the computer via a second channel of the ADC.

To power each robot, a six cell nickle cadmium (NiCd) battery pack was used that provided 7.2V when fully charged. The pack conveniently nestled on top of the front deck of each robot right in front of the citadel and underneath the plow mechanism. The robot could operate for more than 1.5 hours on a single charge.

3.2.3 Robot Software

The software that our robots used to implement their controllers was quite simple. The robot controllers were implemented as behaviour based controllers and were patterned after the robot finite state machine that was developed in Chapter 2. Refer to Figure (3.6) for the following description. Just like our model assumes, our robots are normally found in the plowing state. The plowing state is implemented on our robots with a behaviour called PLOW. While PLOW has control of a robot, the robot will simply move in a straight line in whatever direction it was pointing when PLOW gained control. The behaviour of the finishing and colliding program states are implemented by a single behaviour: RE-ORIENT. RE-ORIENT re-orients a robot somewhat randomly by rotating the robot for a set period of time. The orientation of a robot after the RE-ORIENT behaviour times out is somewhat random because the robots use no position feedback when they turn. The SLEEP behaviour is not present in our model. SLEEP halts a robot when the light level in the room falls below some threshold.

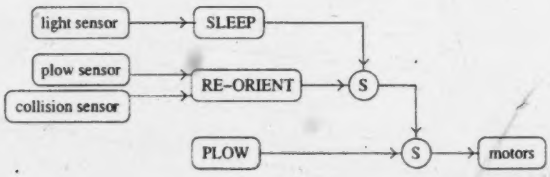


Figure 3.6: This is a behaviour based representation of our robots' control software. The robots have three behaviours: PLOW, RE-ORIENT and SLEEP. The first two behaviours correspond to the three program states. Because the finishing and colliding program states have the same behaviour, we implement them as one behaviour that can be triggered by two different phenomena. The SLEEP behaviour halts a robot when the light level falls below some threshold.

3.2.4 Final Touches

To finish the robots, a few final touches were added to their design A small piece of wood was added to the back of the plows on each side. These are the pieces seen in the bottom of figure (3.4). They extend back far enough from the plow to cover the front of a robot's treads when viewed from the side. They prevented gravel from getting underneath a robot's treads while it turned. Were gravel to get under the treads, the robot would climb up on it and the plow would loose contact with the ground. More gravel would likely get under the robot and could easily cause it to climb the wall of the nest and leave the construction site. The undersides of these two pieces of wood had sweeps installed to reduce their friction with the working surface. All surfaces of the robot except for the treads and electrical components were painted the same colour as the plastic chassis. A piece of paper was formed into a simple cowl that was used to cover the computer (a small hole in the cowl allowed the light

sensor to look upwards). The cowl and paint were added to make any image processing done at a future date easier. The robots could be identified as the large yellow/orange objects in the images.

3.2.5 Compliance with Model

In the last chapter, a theoretical model of our multiple robot system was developed. In order to test this model, our robots should comply with the assumptions made by our model. The theoretical robots wander randomly about the nest plowing gravel and re-orienting themselves when the force on their plow exceeds F_{p_o}. They also re-orient themselves after colliding with other robots. Only the robot that caused a collision should react to it.

The real robots conform to the model quite well. In the model, the position of any robot is completely random at any time. The real robots move in straight lines, but re-orient themselves randomly after a collision or once the force on their plow exceeds its threshold[2]. The real robots were equipped with sensors to allow them to measure the force exerted on them by the gravel and to sense when they collided with other robots. Unless a robot collided head on with another, only one of the robots would detect the collision and react. One of the assumptions of our model was that no two robots would change program state at the same time. Even in the case of a head on collision, one of the robots would detect the collision before the other. Thus the sampling periods of the system could be made frequent enough that, in our experiments, the probability of any two robots changing state in the same sampling period would become insignificant.

3.3 Environment

Our experimental environment consisted of the floor of one of our labs. The gravel used for construction was landscaping rock (dolomite). The average size of a piece of gravel was about that of a chestnut. The gravel came in bags that also contained a substantial amount of dust and smaller rock chips. The rock was filtered over coarse screen to remove the dust and chips before it was used in the experiments. Any pieces with a low profile (less than 5mm or so) were also removed to prevent the robots' plows from getting jammed by rocks wedging themselves underneath them. The area of the floor used was square and approximately 3.5m on each side. To avoid having the boundaries of the environment interfere with our experiments, no external walls were used to enclose the area. Instead, the boundary was simply marked on the ground with adhesive tape.

[2] From one second to the next, the position of a robot depends highly on its previous position. Nonetheless, the position of a robot is relatively random because of the irregularity of collisions with other robots and the randomness of the amount that a robot will turn after a collision or pushing back the nest wall.

3.3.1 Compliance with Model

Our experimental environment follows that assumed by our model. In the model, the gravel is of uniform size. In reality, the sizes of the individual pieces of gravel were relatively constant. The surface on which the gravel was spread was hard, smooth and uniform. Thus the friction that the real gravel exerted on a robot was independent of its location in the environment.

3.4 Running Experiments and Collecting Data

Before a typical experiment began, the gravel was spread evenly about the floor of the environment. For an n-robot experiment, an area of the floor large enough to contain n robots was cleared in the center of the floor. The lights in the lab were turned down enough to make the robots "sleep". The robots were then turned on and placed in the cleared area pointing outwards from the clearing. A camera over the field allowed the experiments to be recorded on video tape. The tape was started and the lights in the lab turned up to start the robots. Refer to Figure (3.7) for an example of the initial state of a four-robot experiment.

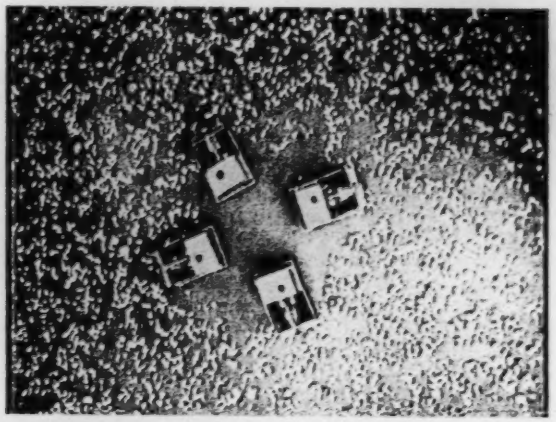


Figure 3.7: Start state of a four robot experiment. The robots are arranged facing outwards from a round clearing in a field of evenly spread gravel. This image was taken with the overhead camera that was used to record our experiments.

The robots were allowed to operate for 30 minutes per experiment. At the end of an experiment, the lights would be turned down and the tape stopped. If at any time during the experiment any of the robots became immobilized or got caught on another robot, the experiment would be paused (using the lights) and the immobilized robots freed and randomly placed in the nest. Similarly, if a robot ever climbed the walls of the nest and left the construction site, the experiment would be paused and the robot returned to a random

position in the nest. The total time for which an experiment had been paused would be added to the end of the experiment to ensure that the robots operated for a total of 30 minutes.

Three trials of each system configuration were run. Experiments were run with one, two, four and six robots. In four trials, a robot ran out of power or became damaged. In these few situations the experiments were cut short (ending at 24, 27, 28 and 29 minutes). This was not seen as a problem since the nest's construction had typically slowed significantly by the time that any experiment had to be cut short.

3.4.1 Making Measurements

The camera that was mounted over the field recorded the entire construction process. Our theoretical model of our system assumed that a circular nest would be produced of a certain radius. Thus the radii of the areas actual nests and their shapes were our two variables of interest. Shape was qualitatively assessed. To measure the area of the nest over the course of the experiment, images were grabbed from the video tapes of the experiments every three minutes starting with $t = 0$. Using a paint program, the cleared area was filled in with a colour not found in the rest of the image. The number of pixels of that colour were counted. A sheet of paper of known area was photographed in the field to calculate the number of pixels per square meter. Distortion due to the camera's lens was ignored.

3.5 Results

The remainder of this chapter will present the data gathered from the multiple robot experiments that were just described. In the next chapter the experimental results will be discussed and compared to the behaviour predicted by the models.

A total of 12 experimental trials of 30 minutes each were conducted and video taped. Each of the four distinct configurations were tested three times. Figure (3.8) shows the area that was cleared for each trial.

The average area cleared for each configuration is shown in figure (3.9). In our model, the radius of the nests cleared was the dependent variable of interest, not the nest's area. It was also assumed that the shape of the nest cleared would be circular. Setting the areas from figure (3.9) equal to πr_n^2 and solving for r_n yields Figure (3.10).

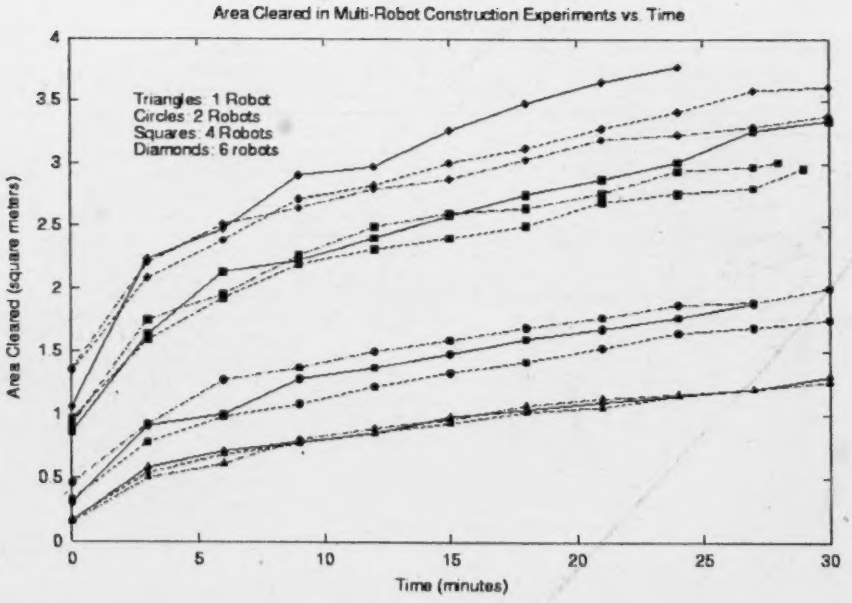


Figure 3.8: Areas of the nests cleared in experiments with various robot populations. Here we can see the behaviour of the various robot populations with respect to their nests' growths.

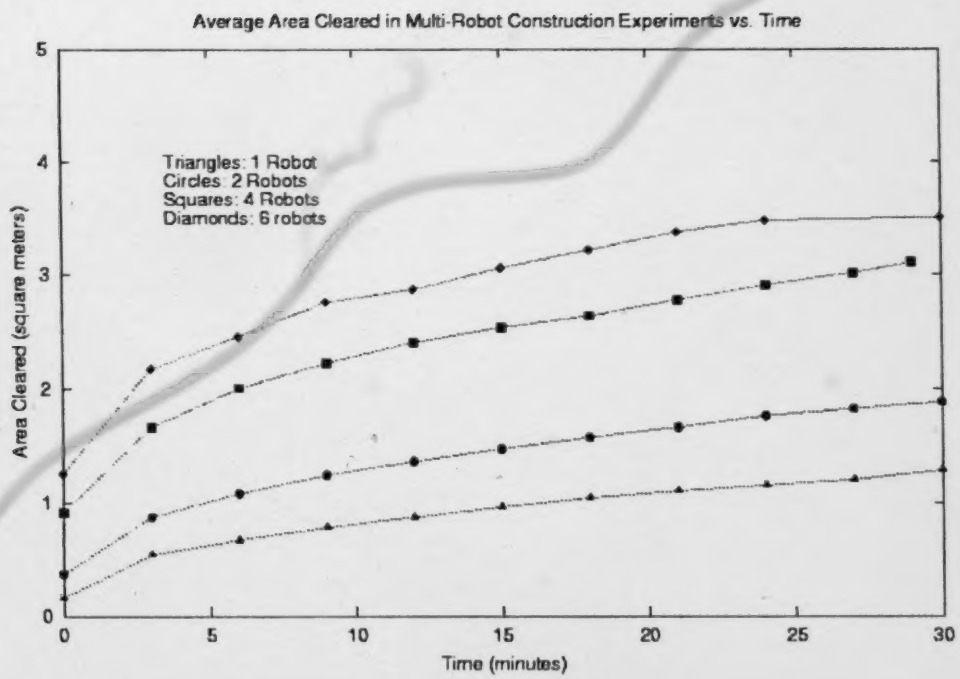


Figure 3.9: Average area cleared for all experimental configurations. This graph depicts the average growth of the various nests for each experimental configuration.

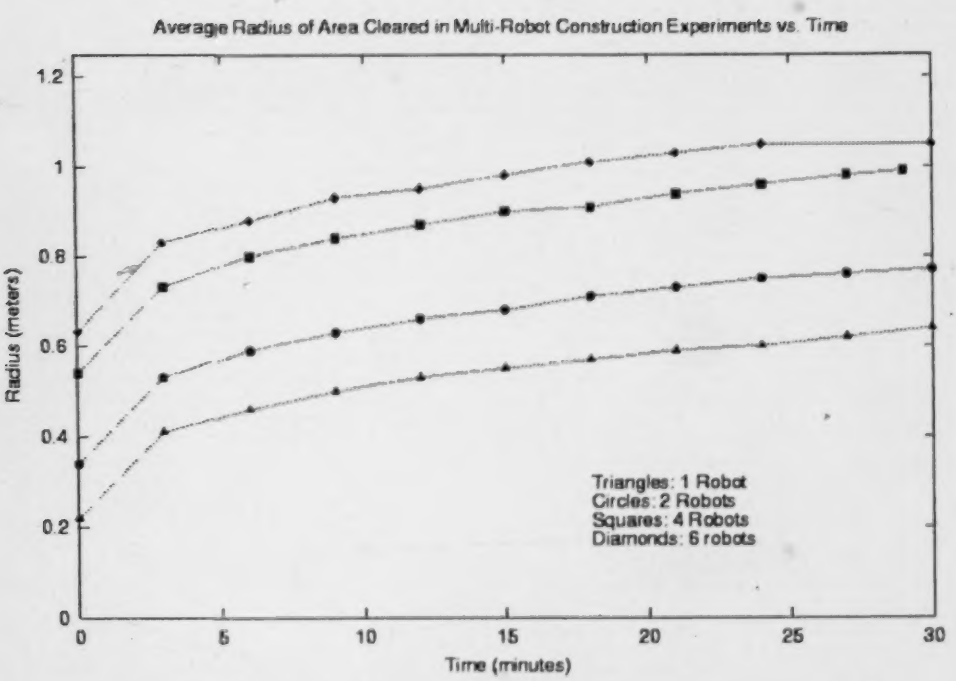


Figure 3.10: Radius of nest cleared for all experimental configurations. This graph was produced by assuming that the nests that our robots cleared were circular and solving for the nests radii. This is the type of data which our model produces.

Chapter 4

Discussion

4.1 Introduction

This is the final chapter of this thesis prior to our general conclusions. We will describe how the model of Chapter 2 was applied to the experiments that we conducted. Discrepancies between our model's predictions and our experimental results will be identified and explained. Following that, we will discuss the effects of changing the robot population of an experiment and changing the plow threshold of our robots. The similarities between our system and other collective robotic systems will be discussed. Finally, we will outline future research steps by returning to our example of a mission to Mars.

4.2 Modeling the Experiment Configurations

At this point, a general model of our multiple robot construction system has been developed and experiments with real robots have been conducted to verify our model. Now, the model's predictions will be compared with the experimental data. Before we can do this, we must set our model's various parameters to match our experimental environment. The parameters that need to be measured are r_r, $\dot{w}_r$, s_o, τ, F_{p_o}, F_g and ρ_o. Our robots were not circular like their counterparts in the model. The robots in our model were assumed to be circular to ease the calculations of the probabilities of collision and pushing against the nest wall. In those calculations, the area of the footprint of a robot was what really mattered. Assuming that the robots were circular allowed the area of the robots to depended only r_r. The actual robots were rectangular, measuring 40cm by 23cm. Thus their area was 0.0092 m^2. The radius of a circle of the same area is 0.17m. This is the value that was used as r_r when modeling. The width of the plowing mechanism of our robots was 0.2m. s_o is the speed of a robot in the plowing state of its program. This was measured as approximately 0.1m/s. The model assumed that a robot would remain in the finishing or colliding states of its program the some length of time. The unit of time used when modeling the experiments was the second. τ was measured to be two seconds.

F_{p_o}, F_g and ρ_o were a more difficult to specify. The gravel in our model was composed of identical pieces of gravel. The real gravel does not live up to this expectation. The density of gravel, ρ_o, varies slightly over the environment. Although the gravel was carefully spread out, it would have been impractical to distribute it perfectly. These three parameters were specified as follows. An estimate was made of F_g and ρ_o. F_{p_o} was set based on the guess of F_g and the average number of pieces of gravel that robots were seen to be plowing just before switching to the finishing state of their programs. We used the average starting radius of the two-robot experiment as r_o. F_{p_o}, F_g and ρ_o were adjusted until the early growth of the nest according to the model agreed with the early growth seen in the two-robot experiments[1]. The same F_{p_o}, F_g, ρ_o, r_r, τ, s_o and w_r were used for all of our modeled results.

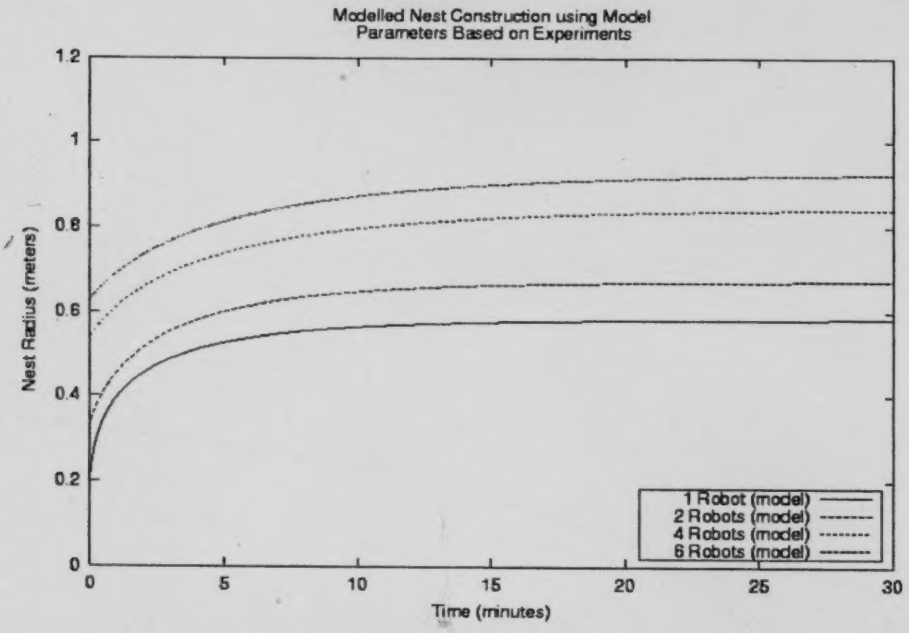


Figure 4.1: This graph shows the nest growth predicted by our system model for our experimental conditions. The average starting radii for each robot population were used as the initial radii for the modeled curves.

The starting radii of the experiments were used as the starting radii for the corresponding modeled results. Figure (4.2) shows the predicted construction of the nest for each experimental configuration. In the next section the model and the experimental results will be compared and discussed.

[1]The two-robot experiments were used to set F_g, ρ_o and F_{p_o} because they gave the most repeatable results of all of our multiple robot experiments.

4.3 Comparison of Modeled and Actual Results

The model developed in Chapter 2 made several simplifying assumptions that were not satisfied by our experiments. What are the discrepancies between the model's assumptions and the our experiments and how do they affect the model's behaviour relative to that of the systems of the experiments? The data from our experiments and from the model's can be seen in Figure (4.2).

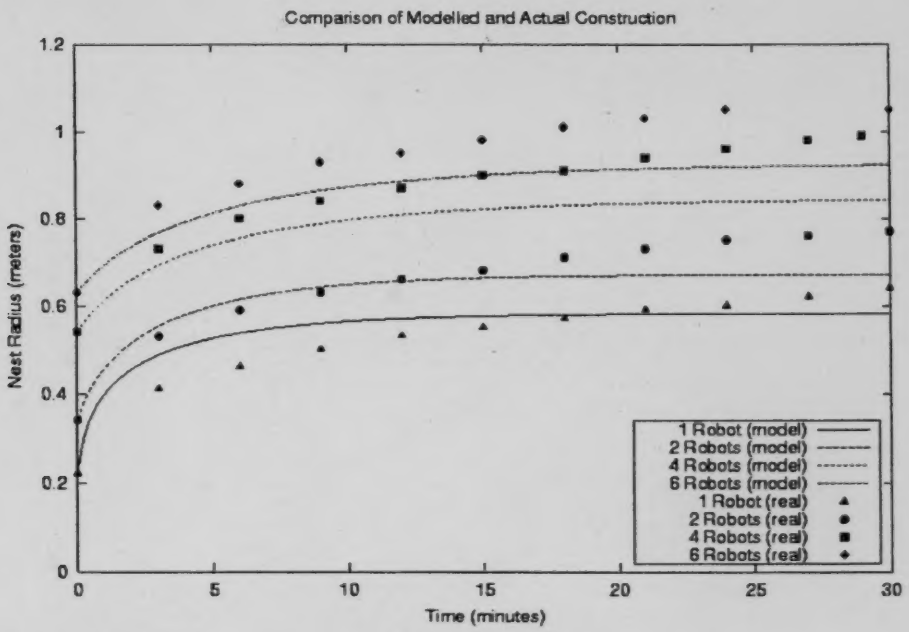


Figure 4.2: This graph shows both the data from our experiments and our model's predictions of those experiments.

The model and the experimental data agree with each other quite well. Just as the model predicts, the increase in the radius of the nest is large at first, and then slows quickly. Much of the growth of the nest occurs in the first 15 minutes of the experiments. The final size of the nest appears to depend more on the initial size of the nest and not on the robot population taking part in its construction. The model captures the essence of the real nests' growth.

4.3.1 Discrepancies Between the Model and the Experiments

From Figure (4.2), it is obvious that there are certain aspects of the real system that are not included in the model. These discrepancies will now be addressed one by one.

Steady State of the System

In our experiments, the nest sizes did not settle to a constant size after 30 minutes as the model predicted they should have. Instead, they appear to have grown at a constant, low rate after their initial quick growth. The robots in the model were capable of measuring the force that the gravel exerted on them. The real robots could measure this force too, but not as well. When the robots were designed, their plows were designed as low frequency sensors. This means that the plows cannot detect rapid changes in the forces applied to them. There is a slight delay between a force being applied to the plow sensor and the sensor detecting it. The effect on our system is that for a short time after a robot comes into contact with the wall of the nest, it will attempt to push it back even if the wall opposes the robot with a force greater than F_{p_o}. The nest wall won't extend back forever, though. It will continue to get stronger until it presents a force to the robots greater than the friction between the robots' tracks and the floor. At this point, a robot will be unable to move the wall regardless of any delay in its reaction to the force of the wall. Were it necessary, we could predict this phenomenon by including the frequency response of the plow sensor and the friction of the robots' tracks on the floor in our model. This phenomenon, however, is minimal. It is also more of a hardware issue than a theoretical one. It would be better addressed by improving the robots' minimal sensory apparatus.

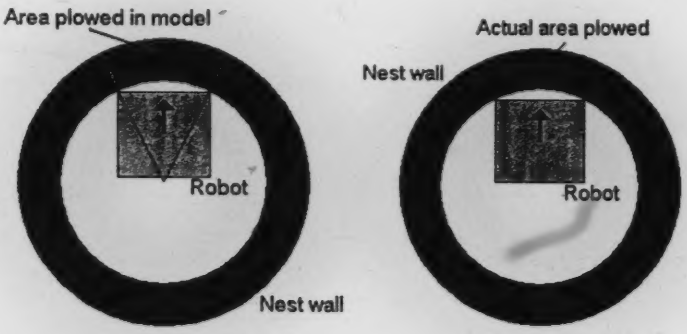


Figure 4.3: Comparison of area of the nest wall that the model assumes is plowed (left) and the area of the nest wall that is actually plowed (right)

Strength of the Nest Wall

The strength of the wall in our model is likely an over estimate. This is due to simplifications made in the model's derivation. In the real system the gravel had a finite size per piece. Thus the real gravel occupied a finite area. The model assumed that the gravel that had been incorporated into the wall of the nest could be found right at the edge of the cleared area. In other words, the model assumed that the physical size of each piece of gravel would

be zero. Because real gravel takes up space, the wall of the nest had a finite thickness unlike the wall in our model, which had zero thickness. When a robot pushed against the nest wall, it pushed on a section of the inner edge of the wall of width w_r. The area of the wall that the model assumed that a robot would push against was bounded by the inner and outer edges of the wall and the two radii of the nest that passed through the outer points of contact of the plow on the inner edge of the nest wall (Figure (4.3), left). In reality, a robot would push on the gravel in the area bounded by the inner and outer edges of the wall and the lines perpendicular to the plow and through the outer points of contact of the plow with the wall (Figure (4.3), right). This second area is smaller than the first and thus presents less resistance to the robots.

An important result of Chapter 2 was that the nest construction process would be self terminating. That is, the nest would grow to a certain size and then grow no more regardless of whether the robots continued to operate of not. If the force that the nest wall opposed a robot with was less in reality than was assumed by our model, will the system still self terminate? All that needs to be satisfied for a nest's construction to self terminate is that the thickness of the nest's wall increases with the nest's radius as this would cause the resistance of the wall increase with nest's radius too. Although it is the robots that push against the wall of the nest, it is the outer edge of the wall that sweeps more gravel into the wall. If the radius of the outer edge of the wall was taken as r_n, it would still move outward when the robots pushed against the nest wall[2]. It has already been shown that the amount of gravel in the nest wall will increase as it is plowed outwards (see Section 2.1.3 in Chapter 2). The nest wall's thickness will increase whether the inner or outer edge of the wall is used as the nest's radius. Thus regardless of whether a robot pushes on the section of the wall shown in the left or right of Figure (4.3), the construction will still self terminate.

Uniformity of the Nest Wall

The assumption that the nest wall would be of uniform strength was also inaccurate. The assumption that the nest's wall would be of uniform strength along its length was made because it simplified the statistics used in analysing our system as the nest wall's strength became a function of the current and initial nest radii. The strength of the nest's wall at a given point on the wall depends on how far that section of the wall has been pushed outwards from the initial center of the nest. Over time, each section of the wall will be pushed against the same number of times as every other section. Early in the construction process, though, the nest wall will have varying strength over its length. The question of the nest's shape will be discussed later.

[2]At the beginning of the construction, the inner and outer radii of the nest wall are the same. Since the wall cannot get thinner than zero thickness, the outer edge of the nest must always expand outward with nest at least as quickly as the inner edge of the wall.

Early Growth of the Nest

If the uniform wall assumption does not hold in the early stages of construction, how will the actual early growth of the nest differ from the model's predictions? When a robot first pushes against the nest wall, the wall will be very weak. The robot will plow further into the nest wall on this first plowing than it will on any other. When the wall presents little resistance to a robot, the robot will plow deep into it. The wall could be considered *soft*. The key to early nest construction is what happens when a robot switches to the finishing state for the first time. At this point, it is deep into the field of gravel. Refer Figure (4.4) for this discussion. Now the robot must get back into the central nest area so that it can plow against some other part of the nest wall. The problem is that the robot cannot simply rotate since it is bounded by gravel at its sides.

Figure 4.4: A robot embedded in a soft section of the wall. This robot has plowed deep into the nest wall and will now be prevented from simply rotating to re-orient itself. It will likely take several attempts for this robot to free itself from this soft wall.

It is much like trying to free a car that is stuck in a snow bank. One cannot simply rotate the car as it is partially imprisoned by the channel that it made when it entered the bank. With the algorithm that we used, the robot will still try to rotate. When the robot stops turning (the amount of rotation is determined by a *time delay*, not by an angle), it will more likely than not be facing part of the pile of gravel that it made seconds before. The robot will encounter more resistance from the nest wall on its next push than the model predicted. The model assumed that the gravel that the robot has just plowed would be spread evenly throughout the nest wall with no localized strong points. The real robot will

encounter a stronger section of the wall than the model predicted and it will not expand the nest as much as the model had predicts it would on its next push. If there are only few robots, very little plowing will have occurred before they all get bogged down by soft sections of the nest wall.

With larger populations the story changes slightly. When each robot is watched individually the situation appears to be the same as before. Each robot will still see a soft nest wall on its first push. Therefore each robot will add roughly the same area to the nest by the time that it switches to the finishing state. The nest will expand by an amount equal to n "first plowings" in an n-robot system in the first n plowings (one per robot) of the system. Our model allows only one "first plowing", though. Each subsequent plowing increases the nest size less and less. In the first few minutes of construction, the difference between subsequent plowings is quite high according to our model. Thus a real n-robot system will be somewhat sped up in the first moments of the construction process when compared with the model. The greater n, the greater the discrepancy between the model's predictions and the actual system's growth. Of course, a multiple robot system will experience robots getting temporarily stuck in the soft sections of the nest wall early in the nest construction. With smaller populations, getting stuck in the walls should dominate the early growth of the system because the robots will be slowed by soft walls before much plowing has been accomplished. With larger systems, the effect of having more "first" plowings will dominate. This is seen in Figure (4.2). The one and two-robot systems grow more slowly at first than the model predicts that they should, while the four and six-robot systems grow faster than they should.

Construction in the Colliding State

Both the finishing and colliding states of a robot's program delay it from doing further work on the nest according to the model. The primary difference between these two states is that a change of state from plowing to finishing was assumed to be the only time that nest construction occurs. In the model, it was only at this change of state that the nest was modified in any way. It was often observed that the real robots would carve out new nest area with their corners as they rotated in both the colliding and finishing states. With an increase in robot population comes an increase in the probability of collision. This increased probability of collision increases the amount of this secondary construction that occurs. In our model, the robots were assumed to be circular. If the real robots were circular and rotated about their centers, they would not modify the nest while rotating.

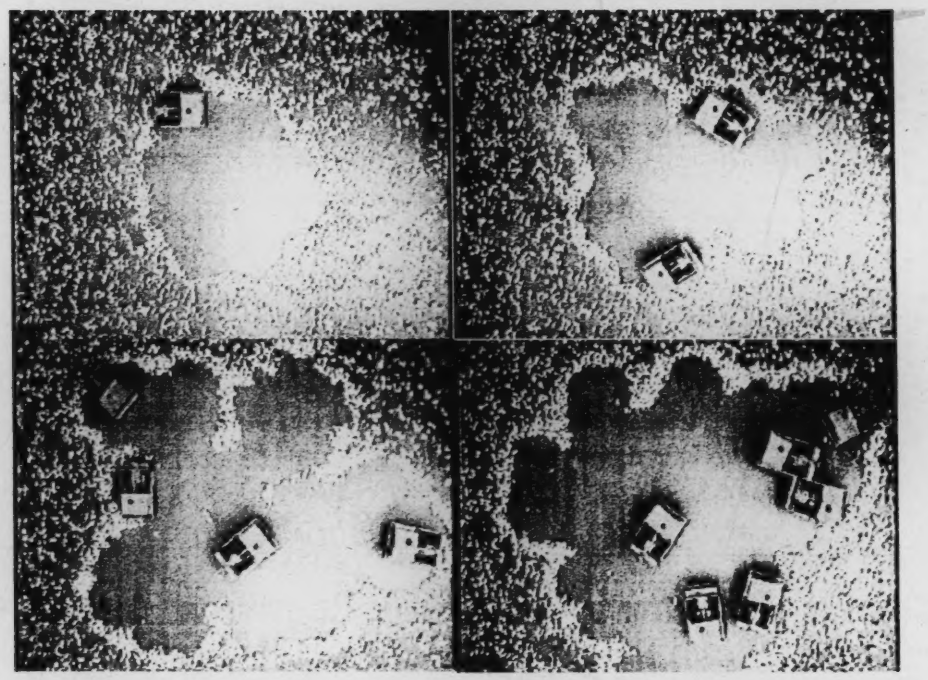


Figure 4.5: Here we see final nests from each of the four different robot populations that we experimented with. Note that by the end of the experiments, the nest walls are relatively uniform in thickness. Also note that the nests are not circular. This means that their walls are longer (and thus weaker per unit length) than our model predicted.

Nest Shape

The final discrepancy between our model's expectations and our real system is the shapes of the nests produced. See Figure (4.5) for images of the final nests produced by several of our experiments. Although these nests could be described as "round", they are not circular. A circle has the smallest perimeter for the area that it encloses when compared with all other shapes. The model assumes that the nest will have this shape. In other words, the model assumes that all of the gravel that has been cleared is packed into the shortest possible nest wall. This means that the nest walls of our model are as strong as possible. Were the perimeter of the nests longer for a given area, the amount of gravel per unit length of the wall would be less. The less circular the nest in the real experiments, the weaker the wall of the nests would be when compared with our model. Most of the experiments saw robots producing bulges in their nests like those seen in Figure (4.5). When a robot broke through the nest wall and started to clear out a bulge, it, in a sense, created a small satellite nest. This nest will grow quickly at first just like the main nest did. The growth of the satellite nest will be included when we measure the main nest's growth. Therefore the main nest will grow quickly for a short period of time after a new satellite nest has been started. If these satellite nests appear on a somewhat regular basis and grow to roughly the same size, they will add a constant amount of area to the nest per unit time. This may account for the relatively constant, positive growth that the actual nests experience after the initial period of rapid growth ends.

4.3.2 Summary

Despite the discrepancies between our model's predictions and our experimental results, the model is still very useful. The aim of the model was not to predict the growth of our specific system exactly. Instead, it characterizes the growth of this type of multiple robot construction system in general. Our robotic system was composed of *very* simple robots. When the main sources of error in the model were identified, many of them were the result of assumptions made by our model that were not satisfied by our actual robots. If it were necessary, the real robots could be modified to comply better with the model or the model could be modified to include more aspects of the real robots. Regardless, the model gave us insight into the behaviour of the system. Our experiments demonstrated the effects of not satisfying some of the model's assumptions. The model's general accuracy allows it to be used to predict the effects on the system of modifying various system parameters.

4.4 Effect of Robot Population and Plow Threshold on Nest Construction

In a realistic setting, there are only two variables which we could set to vary the effectiveness of our collective robotic construction system. These variable are the robot population size and the robots' plow thresholds. All of the variables are dictated to us by the robots that we use (s_o, w_r and τ) and the environment in which we want to use them (r_r, F_g and ρ_o). In Chapter 2, we derived equations and a model that told us that more robots would build a nest faster than fewer and that increasing F_{p_o} would increase the equilibrium size of a nest. Are there limits to these gains?

4.4.1 Effect of Population Size

It makes sense that more robots will build a nest faster than fewer robots. An n-robot system will get to its m^{th} instance of incremental progress sooner than an $(n-1)$-robot system because the probability of a robot switching to the finishing state in the n-robot system will be greater. Of course, because the nests of both of systems asymptotically approach their final size, they will have roughly the same size after some period of time if they had the same initial size. The larger the population for a given nest size, the higher the probability of robot-robot interactions will be. The more likely a robot is to encounter another robot at any given instant, the more careful it has to be while in the finishing or colliding states. In our experiments, the robots simply backed up and turned when they entered either of these states. While in these states, they did not check for collisions with other robots. This strategy started to break down at six robots. Robots would switch to the finishing or colliding state and then ram other robots while reversing. Often this would cause the two robots to get hung up on each other or one of them would get pushed onto the nest wall and become stranded. Interference between robots is an important problem in multiple robot systems that has yet to be solved [10]. In general, the $(n+1)^{th}$ robot will increase the productivity of the system less than the n^{th} robot did because the addition of the $(n+1)^{th}$ robot will increase the probability of a collision more than it did the probability of incremental progress occurring.

4.4.2 Effect of Plow Threshold

One would likely make the robot population of a system large enough to get the maximum productivity out it. The only other variable to modify would be F_{p_o}. Figure (4.6) shows the relationship between the final nest size and F_{p_o}.

Increasing F_{p_o} will increase the final nest size because a robot with a greater F_{p_o} can push back a stronger wall. Each construction step will be larger too, since each robot will be able to plow more gravel before it switches to the finishing state. It would seem logical to

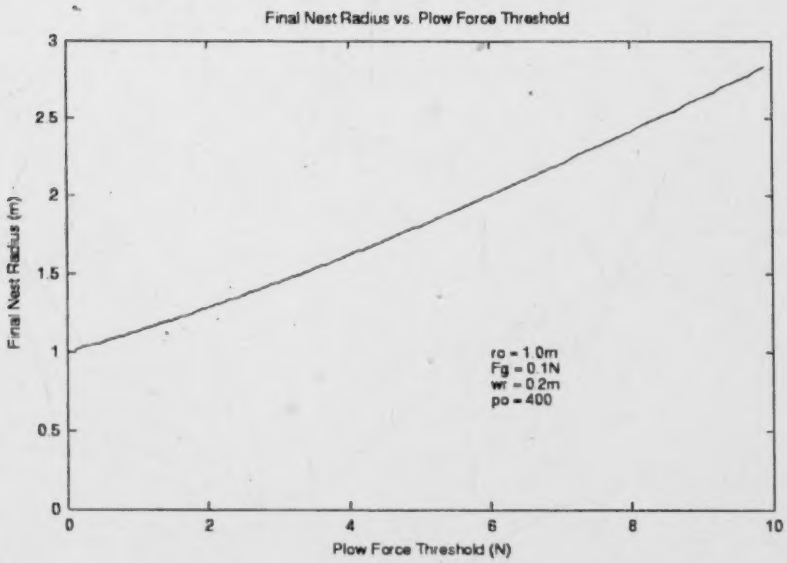


Figure 4.6: This graph shows the effect that increasing the robots' plow thresholds, F_{p_o}, would have on the nest's equilibrium size. If F_{p_o} is made too large, the robots will end up stuck in the nest wall.

make F_{p_o} as large as possible. Unfortunately it is not as simple as this. Recall the analogy drawn earlier of a car stuck in a snow bank. Increasing F_{p_o} would put our hypothetical car further into the snow and would make getting it out more difficult. Similarly, a robot with higher F_{p_o} will drive further into the nest wall when it finally switches to the finishing state and tries to get back to the main nest area. It is possible for such a robot to maneuver back to the nest, although more care would be needed in the finishing state to avoid getting stuck. Thus F_{p_o} should be made as large as a robot's maneuverability can accommodate or large enough to clear a nest of the desired size, whichever is less.

4.5 Similarities of our System to other Multiple Robot Systems

Our robots construct a nest through the interaction of two basic behaviours. These behaviours are activated by collision sensors and a force sensitive plow. Remember, our robots have only *two* behaviours[3]. The finite state machine representation that we used in Chapter 2 had three states because the RE-ORIENT behaviour (Figure (3.6) signaled different phenomena in a nest depending on how it was triggered. Other multiple robot systems have employed similar behaviours. For example, the box pushing robots of [17] had a behaviour called AVOID that performed much the same function as our RE-ORIENT behaviour. The box pushing robots had a total of five behaviours. Does this mean that box pushing is a more complex task than our type of collective construction? Perhaps. In our mission to Mars example, the robots would likely encounter rocks that were too large for a single robot to move on its own. In this case, several robots would have to cooperate in order to move the rock. One could imagine an incarnation of our construction system in which the controller of the box pushing robots of [17] would be activated as a single behaviour in order to move a large object. The important issue here is that a multiple robot system is defined by more than its basic behaviours. It is the interaction of the environment in which the system is deployed and the interconnections of the behaviours of the robots in the system which sets the global system behaviour.

4.6 Next Steps

Having implemented and studied a simplified version of the construction algorithm used by the ants of [8, 9], what are the next steps? Let us return to our hypothetical mission to Mars in which a swarm of robots would clear a landing pad for a future mission. We would want our robots to be able to find a suitable site to clear and, more importantly, agree on where to begin construction. If 20 robots were sent and instead of one large clearing we got

[3]We do not count the SLEEP behaviour as it was added to make running experiments easier.

20 smaller clearings, the system would have failed. The robots need to be able to find and agree upon a suitable nest site.

We would also like our robots to build larger nests. The only way to do this would be to increase F_{p_o}[4]. The main problem with this strategy would be that robots would get stuck in soft portions of the nest wall. A wall is considered "soft" if its strength is much less than F_{p_o}. As a nest enlarges, its wall will get stronger. If the robots used their plows to measure how soft the walls were, they could set F_{p_o} appropriately. As long as the robots had sufficient pushing power (motor torque and traction), they would be able to continually expand the nest by increasing their F_{p_o}'s. Once they stopped increasing their F_{p_o}'s, the nest size would stabilize.

Finally, the issue of global perception could be addressed. Currently, our robots have no concept of the size of their nest. It is up to an outside observer to stop the system. Our robots could indirectly measure the size of the nest that they were building by keeping track of how often they collided with other robots. As the nest enlarged, the rate of collision would decrease. In this fashion, the robots could estimate how larger their nest was. This would be especially valuable in a mission to another planet. The robots sent would likely be used to perform other tasks other than construction. Once the nest had reached a desired size, the robots would be able to divert their attention to other projects.

4.7 Summary

In this chapter, we compared the results of our experiments to the predictions made by our model. The discrepancies between the two were discussed. Many of them had to do with shortcomings of our robots. Regardless, the model captured the general behaviour of our system and was used as the basis a discussion of the effects that changing various system parameters would have on the system. We looked at how the population size and the plowing threshold of the robots would change the nest's growth. We also discussed our system's similarities to other multiple robot systems. Finally, logical next steps in this research were discussed using our now familiar mission to Mars as an example.

[4]The robots would be stuck with whatever sized initial nest they could find. In other words, increasing r_o is not a realistic option

Chapter 5

Conclusions

5.1 What Have We Done?

In this thesis, an algorithm for collective robotic construction was presented. Several methods were used to predict the collective construction system's behaviour. Following our theoretical study, the algorithm was implemented in the laboratory using real robots. We discussed the performance of the actual system and the predictions that were made by our models. This research is important because it represents a new stage in collective robotic research.

The construction algorithm that we implemented was originally observed in a species of ant as the ants built their nests. It was identified as a good candidate for laboratory implementation because the nests that the ants built were two dimensional. This meant that any robots used to test the algorithm would only need to be able to move in two dimensions. This considerably simplified the mechanical design of the system and the analysis of the system's behaviour.

Our theoretical treatment of our construction algorithm introduced a new approach to predict a system's behaviour. This approach used a statistical model to generalize a temporal system model of our system. We believe that this method represents a new technique to model a multiple robot system's global behaviour.

Experiments were carried out using real robots. The robots were custom built for this research. Several different experiments were run out to examine the effects of population size on the system's behaviour.

The experiments yielded consistent results. The model that was developed behaved similarly to our implemented system. Discrepancies between the predictions made by our model and the outcomes of our experiments were identified and explained. Finally, future directions for research were identified using our hypothetical robotic mission to Mars to motivate the discussion.

5.2 What is Special about this Research?

To our knowledge, this is the first time that collective construction has been implemented with actual robots. The experiments in this thesis involved robots which actively modified their environment in an orderly and controlled fashion.

Modeling collective systems is a difficult task. Many of the microscopic properties of the system (individual robot velocities, etc.) are not relevant when the global behaviour of the system is what is being studied. The model of our system was developed using an approach involving Markov chains which we believe is unique.

5.3 Why is this Important?

The research carried out in this thesis is important for several reasons. Collective robotics is still a relatively young field of research. Any successful implementation of a system of collective robots further demonstrates the feasibility of this type of robotic solution. Researchers in this field have always claimed that collective robotic systems can adapt to a changing environment. Our experiments not only took place in a changing environment, it was the robots themselves that were modifying their environment.

Modeling the global behaviour of a collective systems has always been a critical problem with their study. We presented a new technique to analyse our system. This model used the individual robot behaviours to predict the global system behaviour. Creating a tangible link between local and global behaviour is a good step toward understanding how to design these systems.

Bibliography

[1] Agassounon, W. and Martinoli, A. (2002). Efficiency and Robustness of Threshol-Based Distributed Allocation Algorithms in Multi-Agent Systems. In *Proceedings of AAMAS 2002, July 15-19, 2002, Bologna, Italy.*

[2] Anton, H. and Rorres, C. *Elementary Linear Algebra: Applications Version, 7th ed.* John Wiley & Sons, inc. Toronto, 1994.

[3] Beckers et al. (1994). From Local Actions to Global Tasks: Stigmergy and Collective Robotics. In *Proceedings of the Fourth International Workshop on the Synthesis and Simulation of Living Systems* (p. 181-189).

[4] Beni, G. and Wang, J. (1989). Swarm Intelligence. In *Proc. of the Seventh Annual Meeting of the Robotics Society of Japan, Tokyo, Japan, 1989* (p. 425-428).

[5] Bonabeau, E. et al. (1999). Swarm Intelligence: From Natural to Artificial Systems, Oxford University Press, New York, 1999.

[6] Brooks, R. (1986) A Robust Layered Control System for a Mobile Robot. *IEEE Journal of Robotics and Automation, 2(1), March 1986* (p. 14-23).

[7] Deneubourg et al. (1991). The Dynamics of Collective Sorting: Robot-Like Ants and Ant-Like Robots. In *Proceedings of SAB '90: 1st International Conference on Simulation of Adaptive Behaviour* (p. 356-365).

[8] Franks, N. et al. (1992) Self-Organizing Nest Construction in Ants: Sophisticated Building by Blind Bulldozing. *Journal of Animal Behaviour, vol. 44, 1992* (p. 357-375).

[9] Franks, N. and Deneubourg, J. (1997). Self-Organizing Nest Construction in Ants: Individual Worker Behaviour and the Nest's Dynamics. *Journal of Animal Behaviour, vol. 54, 1997* (p. 779-796).

[10] Goldberg, D. and Materic, M. (1997) Interference as a Tool for Designing and Evaluating Multi-Robot Controllers. In *Proc. National Conf. on Artificial Intelligence (AAAI), Providence, RI* (p. 637-642).

[11] Hayes, A. (2002) How Many Robots? Group Size and Efficiency in Collective Search Tasks. In *Proc. of the 6th Int. Symp. on Distributed Autonomous Robotic Systems DARS-02, June, Fukuoka, Japan* (p. 289-298).

[12] Hölldobler, B. and Wilson, E. (1994). *Journey to the Ants: A Story of Scientific Exploration.* Belknap Press/Harvard University Press.

[13] Hölldobler, B. and Wilson, E. (1990) *The Ants.* Belknap Press/Harvard University Press.

[14] Ijspeert, A., et al. (2001) Collaboration Through the Exploitation of Local Interactions in Autonomous Collective Robotics: The Stick Pulling Experiment. *Journal of Autonomous Robots, Vol. 11, No. 2, 2001* (p. 149-171).

[15] Jones, J. and Flynn A. (1993). *Mobile Robots: Inspiration to Implementation.* A K Peters, Wellesley, MA.

[16] Johnson, S. (2001). *Emergence: The Connected Lives of Ants, Brains, Cities, and Software.* Simon and Schuster, New York.

[17] Kube, R. and Zhang, H. (1994) Collective Robotics: From Social Insects to Robots. *Adaptive Behavior, vol. 2, no. 2, 1994* (p. 189-218).

[18] Lerman, K. et al. (2001) A Macroscopic Analytical Model of Collaboration in Distributed Robotic Systems. *Artificial Life 7:4, 2001* (p. 375-393).

[19] Lerman, K. and Galstyan, A. (2002). Mathematical Model of Foraging in a Group of Robots: Effect of Interference. To appear in *Autonomous Robots.*

[20] Lerman, K. et al. (2001). *A General Methodology for Mathematical Analysis of Multi-Agent Systems.* USC Information Sciences Technical Report ISI-TR-529.

[21] Lerman, K. (2001). *Design and Mathematical Analysis of Agent-Based Systems.* Lecture Notes in Artificial Intelligence (LNAI 1871), p. 222 ff. Springer-Verlag, Berlin Heidelberg.

[22] MacKenzie, D. et al. (1997). Multiagent Mission Specification and Execution. *Autonomous Robots 4(1), 1997* (p. 29-52).

[23] Materic, M. et al. (1995). Cooperative Multi-Robot Box Pushing. In *Proc. IROS-95, Pittsburgh, PA, 1995* (p. 556-561).

[24] Melhuish, C. et al. (1998). Collective Sorting and Segregation in Robots with Minimal Sensing. *5th International Conference on Simulation of Adaptive Behaviour, Zurich, 1998.*

[25] Sendova, A. Faculty Web Page of Dr. Ana Sendova at the University of West England [Online] http://www.cems.uwe.ac.uk/ absendov/sendova2.gif. Sept. 1, 2002.

[26] Stewart, W. J. (1994). *Introduction to the Numerical Solution of Markov Chains.* Princeton University Press, Princeton, New Jersey.

[27] Theraulaz, G. and Bonabeau, E. (1995). Coordination in Distributed Building. *Science, vol. 269, August 4, 1995* (p. 686-688).

[28] Theraulaz, G. and Bonabeau, E. (1995). Modelling the Collective Building of Complex Architectures in Social Insects with Lattice Swarms. *Journal of Theoretical Biology, vol 177, 1995* (p. 381-400).

[29] Wessnitzer, J. and Melhuish, C. (2002). Building Adaptive Structure Formations with Decentralized Control and Coordination. Poster presentation at *Conference on Simulation of Adaptive Behaviour.*

Appendix A

Markov Chain State Transition Probabilities

In this appendix, we will develop the five possible system state transition probabilities for our Markov chain representation of an n-robot collective construction system.

A.1 Probabilities of Basic Phenomena Occurring in an n-Robot System

There are two basic phenomena that can occur in an n-robot system. These phenomena are that a robot will push against the nest wall or that a robot will collide with another robot. Both of these phenomena are accompanied by a change in a robot's program state.

The probabilities of collision and pushing against the nest wall are based on simple geometric arguments. We assume that the positions of the robots are completely random from one sampling period to the next.

A.1.1 Probability of Robot-Robot Collision

For a robot to collide with another robot, it must get its center within $2r_r$ of the center of the other robot. This will bring the edges of the two robots into contact. If one of the robots takes up a position in the nest that brings its center within $2r_r$ of another robot, that robot will detect a collision. Thus the probability of a specific robot experiencing a collision at any moment is the probability that one of the robots will try to occupy some area within the nest that would bring its center within $2r_r$ of another robot. This probability can be calculated based on simply geometry, as shown in Equation (A.1).

$$P_{c_r} = \frac{\text{Area in nest that would cause a collision}}{\text{Area that robot could occupy}} \tag{A.1}$$

For every robot other than the one being watched for collisions, there is a circular area in the nest of radius $2r_r$ around its center that would result in a collision were the robot of interest to try to occupy it. Thus if there are n robots in the system, Equation (A.1) becomes Equation (A.2).

$$P_{c_r} = \frac{(n-1)\pi(2r_r)^2}{\pi r_n^2} = \frac{4(n-1)r_r^2}{r_n^2} \tag{A.2}$$

The probability of a specific robot not causing a collision, then, is simply $1 - P_{c_r}$.

$$\overline{P_{c_r}} = 1 - \frac{4(n-1)\pi r_r^2}{\pi r_n^2} = \frac{r_n^2 - 4(n-1)r_r^2}{r_n^2} \tag{A.3}$$

If there are n robots in the nest, the probability that none of them will cause a collision is $\overline{P_{c_r}}^n$.

$$\overline{P_c} = \overline{P_{c_s}}^n = \left(\frac{r_n^2 - 4(n-1)r_r^2}{r_n^2}\right)^n \tag{A.4}$$

Finally, the probability of *at least one* of the robots causing a collision is $1 - \overline{P_c}$

$$P_c = 1 - \overline{P_c} = 1 - \left(\frac{r_n^2 - 4(n-1)r_r^2}{r_n^2}\right)^n \tag{A.5}$$

A.1.2 Probability of Pushing Against Nest Wall

The probability of a robot pushing against the nest wall can be found in a similar fashion as was the probability of a robot causing a collision. Let us assume that if a robot comes into contact with the nest wall, that robot will attempt to push it back.

$$P_{p_s} = \frac{\text{Area in nest that would cause a push}}{\text{Area that robot could occupy}} \tag{A.6}$$

For the robot to come into contact with the wall of the nest, it must get its center within r_r of the wall. This will bring the edge of the robot into contact with the nest wall. There is a ring of inner radius $r_n - r_r$ and thickness r_r at the periphery of the nest that would cause a robot to push against the nest wall were the robot to enter it. Thus Equation (A.6) becomes Equation (A.7).

$$P_{p_s} = \frac{\pi r_n^2 - \pi(r_n - r_r)^2}{\pi r_n^2} = \frac{r_n^2 - (r_n - r_r)^2}{r_n^2} \tag{A.7}$$

The probability of no pushing occurring for a specific robot is $1 - P_{p_s}$.

$$\overline{P_{p_s}} = 1 - P_{p_s} = 1 - \frac{r_n^2 - (r_n - r_r)^2}{r_n^2} = \frac{(r_n - r_r)^2}{r_n^2} \tag{A.8}$$

With n robots in the nest, there are n different robots that we must consider for the probability that none of the robots will push against the nest wall.

$$\overline{P_p} = \overline{P_{p_s}}^n = \left(\frac{(r_n - r_r)^2}{r_n^2}\right)^n \tag{A.9}$$

Finally, the probability that *at least one* robot will push against the nest wall is given in Equation (A.10).

$$P_p = 1 - \overline{P_p} = 1 - \left(\frac{(r_n - r_r)^2}{r_n^2}\right)^n \tag{A.10}$$

A.1.3 Basic System Phenomena Summary

We have just calculated the probabilities of the two basic phenomena that could occur in an n-robot system. It was implicit in both of these developments that all of the robots in the nest were capable of detecting these phenomena. In other words, we assumed that all of the robots in the nest were in the plowing program state. Of course, if all of the robots are in the finishing and colliding program states, none of them would react to a collision or contact with the nest wall. In a later section, we will include the number of robots in the plowing program state in these equations when we calculate the system state transition probabilities.

A.2 Including the Duration of Re-orientation

Our robots re-orient themselves in the finishing and colliding program states. This re-orientation takes time. Thus a robot may enter the finishing program state in sampling period t_n and may not switch back to the plowing program state until sampling period t_{n+4}. When a robot re-enters the plowing state depends on how long it takes the robot to re-orient itself. If, while a robot is in the finishing program state, no other robots leave the plowing state, the system will not change state. Thus for each state of the system,

there is some probability that the system will still be in that state in the next sampling period. We define this probability as $P((x,y,z)|(x,y,z))$. Since all possible system state transitions involve a robot entering or leaving the plowing program state, an equivalent definition of this probability is $P(x = i|x = i)$. This probability can also be defined as $1 - P(x = i + 1|x = i) - P(x = i - 1|x = i)$. That is, the probability that the system will remain in the same state for the next sampling period is one minus the probability that the number of robots in plowing state will increase minus the probability that the number of robots in the plowing state will decrease. We will proceed to calculate the probability that the number of robots in the plowing program state will increase in the next sampling period.

A.2.1 Probability of a Robot in the Finishing or Colliding State Switching to the Plowing State

Here, we will calculate the probability that in the next sampling period the number of robots in the plowing state will increase by one. Assume that a robot will remain in the finishing or colliding program state for τ sampling periods after it enters one of these states. If the system is found in a state such that m robots are not in the plowing state, what is the probability that one of the non-plowing robots will switch to the plowing state in the next sampling period?

The easiest way to think of this is to use a time line (see Figure (A.1)). In this case, τ is five. In sampling period two, a robot leaves the plowing state. The time that it will remain out of the plowing state is represented by the five blocks on the time line.

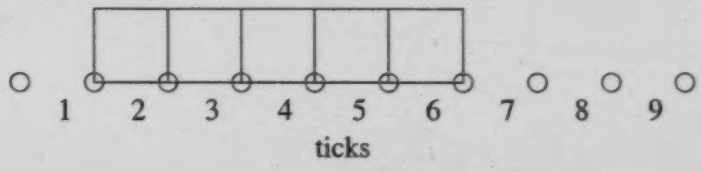


Figure A.1: This is a time line representation of a robot leaving the plowing state. In this example, the robot will remain in either the finishing or colliding state for five sampling periods. The robot left the plowing state in the sampling period two and will re-enter it in sampling period seven.

From the diagram it is obvious that the robot will re-enter the plowing state in the seventh sampling period. If the system was found with one robot in the finishing or colliding state without knowing when that robot left the plowing state (as our Markov chain assumes), it would not be known for sure whether the robot would re-enter the plowing state for certain. With $\tau = 5$, there is a $\frac{1}{5}$ chance that the robot would re-enter the plowing program state in the next sampling period. What if two or more robots are found out of the plowing state? On the time line, this would correspond to multiple overlapping intervals of length τ - each corresponding to a robot in either the finishing or colliding program state (Figure (A.2)). Because each interval is the same length and none of them start at the same time (since no two robots can change state at the same time) none of the intervals will end at the same time. The probability that the number of robots in the plowing state will increase in the next sampling period is related to the number of different ways that the intervals can overlap and to the sum of the length of overlap of each method of overlap. Let the ways in which the intervals can overlap be the *modes of overlap* and the sum of the lengths of the overlap of each mode of overlap be the *total overlap*.

The probability that the next sampling period will see a transition to a system state with one more robot in the plowing state is given by Equation (A.11).

$$P(x = i + 1|x = i) = \frac{\text{modes of overlap}}{\text{total overlap}} \tag{A.11}$$

Like so many other problems in probability theory, this one boils down to counting. The number of modes of overlap of m regions of length τ and their total overlap have to be calculated. There is an iterative method that can be used to accomplish this. Place an interval and number each sampling period of the interval from its beginning until its

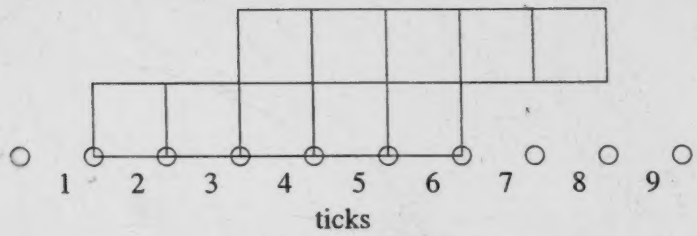


Figure A.2: In this example, two robots leave the plowing state. from sampling period four to six there will be two robots in not in the plowing state.

end starting with τ for the first period and ending with 1 for the final period. The second interval could be placed starting at period $\tau - 1$, $\tau - 2$, ... , 2, 1 if the two regions are to overlap. The placement of the next interval depends on where the second was placed. If the second interval was placed at period $\tau - k$, then the third could be placed anywhere from period $\tau - k - 1$ to period 1 and have the three intervals overlap. This pattern repeats until the m^{th} interval has been placed.

Let each possible placement of an interval be a vertex in a tree. The root of the tree is the first interval that was placed. A vertex in this tree is identified by an ordered pair (i, j). The first number is the sampling period at which the corresponding interval begins. The second number is the number of the interval being placed (the first interval placed is numbered m and the last interval placed is numbered 1). The root of the tree is (τ, m). The rules governing an m-interval overlap tree with interval length τ are given in Equations (A.12) and (A.13)

$$(i,j) \text{ is a vertex of an overlap tree } \rightarrow (1 \leq i \leq \tau) \cap (1 \leq j \leq m) \cap (i \geq j) \qquad \text{(A.12)}$$

$$(a,b) \text{ is child of } (i,j) \rightarrow (b = j-1) \cap (1 \leq a \leq i-1) \qquad \text{(A.13)}$$

Each path from the root of the tree to a vertex with $j = 1$ represents a mode of overlap. The regions are placed at the indices indicated by the i's of all of the vertices on the path. The number of paths from the root to a vertex with $j = 1$ is the same as the number of vertices with $j = 1$. Thus the number of modes of overlap of m intervals of length τ can be computed by generating the appropriate interval overlap tree and counting the number of vertices with $j = 1$. Figure (A.3) shows the 3-interval overlap tree with $\tau = 6$.

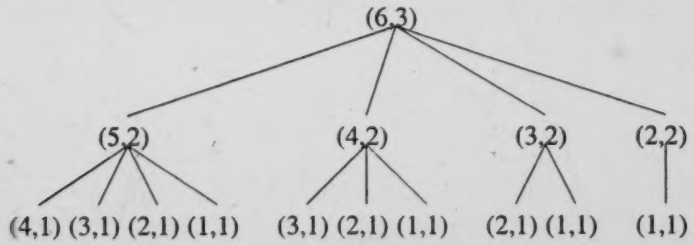


Figure A.3: This tree represents all of the different ways that three intervals of length six could overlap each other. Each path from the root of the tree to a leaf represents a mod of overlap. The first index of a leaf of the tree indicates the amount of overlap for that particular mode of overlap.

Note that the first index of each vertex indicates how much the corresponding interval overlaps the first interval (interval m). Since each interval is placed at least one more sampling period toward the end of the first interval than the previous one, the end of the first interval will come before the ends of the other intervals.

$$\text{modes of overlap} = \begin{cases} m = 1 & 1 \\ m = 2 & \tau - 1 \\ m = 3 & \sum_{i}^{\tau-2} i \\ m = 4 & \sum_{i_2}^{\tau-3} \left(\sum_{i_1}^{i_2} i_1 \right) \\ m = k & \sum_{i_{k-2}}^{\tau-k+1} \left(\sum_{i_{k-3}=1}^{i_{k-2}} \left(\cdots \left(\sum_{i_2}^{i_3} \left(\sum_{i_1}^{i_2} i_1 \right) \right) \cdots \right) \right) \end{cases} \tag{A.14}$$

This means that the amount of overlap of a given mode is just bottom vertex's first index. The total overlap for a given τ and m, then, is simply the sum of the i's of all of the vertices with $j = 1$. In Figure (A.3), there are 10 modes of overlap (10 vertices with $j = 1$) with total overlap of $4 + 3 + 2 + 1 + 3 + 2 + 1 + 2 + 1 + 1 = 20$. Using the pattern that develops in the overlap trees, formulas for the number of modes of overlap and the total overlap become apparent. They are given in Equations (A.14) and (A.15).

$$\text{Total overlap} = \begin{cases} m = 1 & \tau \\ m = 2 & \sum_{i=1}^{\tau-1} i \\ m = 3 & \sum_{i_1=1}^{\tau-2} \left(\sum_{i_0}^{i_1} i_0 \right) \\ m = 4 & \sum_{i_2}^{\tau-3} \left(\sum_{i_1}^{i_2} i_1 \left(\sum_{i_0}^{i_1} i_0 \right) \right) \\ m = k & \sum_{i_{k-2}}^{\tau-k+1} \left(\sum_{i_{k-3}=1}^{i_{k-2}} \left(\cdots \left(\sum_{i_2}^{i_3} \left(\sum_{i_1}^{i_2} \left(\sum_{i_0}^{i_1} i_0 \right) \right) \right) \cdots \right) \right) \end{cases} \tag{A.15}$$

It is relatively easy to write recursive algorithms to calculate both the number of modes of overlap and the total overlap for a given τ and m.

We can now calculate the probability that the number of robots in the plowing program state will increasing in the next sampling period, $P(x = i + 1 | x = i)$. We obtain m from the current system state: $m = y + z$. Calculate the number of modes of overlap using the appropriate form of Equation (A.14). Then calculate the total overlap with the appropriate form of Equation (A.15). Use these two figures to calculate $P(x = i+1|x = i)$ with Equation (A.11).

A.3 System State Transition Probabilities

Having calculated the probabilities of basic system phenomena occurring and included the time that it takes a robot to re-orient itself in our calculations, we are now prepared to calculate our Markov chain's state transition probabilities. From system state (x, y, z), the state of the system in the next sampling period could be $(x + 1, y - 1, z)$, $(x + 1, y, z - 1)$, $(x - 1, y + 1, z)$, $(x - 1, y, z + 1)$ or (x, y, z). We will now address each of these cases.

A.3.1 Probabilities of Robots in Finishing or Colliding States Entering the Plowing State

These two probabilities are the probabilities that the number of robots in the plowing program state will increase in the next sampling period. Since the time required for a robot to re-orient itself is assumed to be the same for both the finishing and colliding program states, the program state that the robot enters the plowing state from depends on the proportion of robots in each of the non-plowing states. For example, if three robots were

in the finishing state and one was in the colliding state and one of these four robots were to enter the plowing state in the next sampling period, the probability of that robot having been in the finishing state would be 3/4. Equations (A.16) and (A.17) provide the formulas to calculate the $P((x+1, y-1, z)|(x, y, z))$ and $P((x+1, y, z-1)|(x, y, z))$.

$$P((x+1, y-1, z)|(x, y, z)) = \frac{y}{y+z} P(x = i+1|x = i) \tag{A.16}$$

$$P((x+1, y, z-1)|(x, y, z)) = \frac{z}{y+z} P(x = i+1|x = i) \tag{A.17}$$

A.3.2 Probabilities of Robots in the Plowing State Entering the Finishing or Colliding States

These two probabilities are the probabilities of basic system phenomena occurring within the system. Earlier, we calculate these probabilities assuming that all of the robots were in the plowing program state. We now include the fact that in a given state of the system, not all of the robots will be in the plowing state. If the system was in state (x, y, z), then there would be only x robots that could possibly enter the finishing or colliding states in the next sampling period. The probability that a robot will enter the finishing program state in the next sampling period is given in Equation (A.18). The probability that a robot will enter the colliding program state in the next sampling period is given in Equation (A.19). In both of these equations, we include the probability that none of the robots that are already in the finishing or colliding program states will re-enter the plowing state. We have to do this because we allow only one robot to change program state per sampling period in our model.

$$P((x-1, y+1, z)|(x, y, z)) = \left(1 - \left(1 - \frac{r_r}{r_n}\right)^{2x}\right) \times \overline{P(x = i+1|x = i)} \tag{A.18}$$

$$P((x-1, y, z+1)|(x, y, z)) = \left(1 - \left(1 - 4(n-1)\left(\frac{r_r}{r_n}\right)^2\right)^x\right) \times \overline{P(x = i+1|x = i)} \tag{A.19}$$

There is a problem with these two equations that is not immediately apparent. It is possible for these probabilities to sum to greater than $\overline{P(x = i+1|x = i)}$. This is because the areas within the nest that would cause collisions and pushing against the nest wall are not mutually exclusive. We solve this problem by scaling Equations (A.19) and (A.18) by their sum divided by $\overline{P(x = i+1|x = i)}$ if they sum to greater than $\overline{P(x = i+1|x = i)}$.

A.3.3 Probability of the System Remaining in the Same State

This probability is $P(x = i|x = i)$. Earlier, we stated that this probability is the same as $1 - P(x = i+1|x = i) - P(x = i-1|x = i)$. We can calculate $P(x = i+1|x = i)$. $P(x = i-1|x = i)$ is simply $P((x-1, y+1, z)|(x, y, z)) + P((x-1, y, z+1)|(x, y, z))$. Thus we calculate $P(x = i|x = i)$ last as given in Equation (A.20).

$$P((x, y, z)|(x, y, z)) = 1 - P((x-1, y, z+1)|(x, y, z)) - P((x-1, y+1, z)|(x, y, z)) - P(x = i+1|x = i) \tag{A.20}$$